Agile Project Management in the Travel Industry

Özkan Murat
Master Thesis Autumn 2016
Agile Project Management in the Travel Industry

Özkan Murat

August 21, 2016
Abstract

Software is used in many areas of our daily lives. Optimal software development processes are essential for successful projects in the software sector which has become a gigantic industry.

Traditional software development processes can result in failure in software projects. Some difficulties exist in traditional processes if they aren’t receptive to the customer’s requirements, or they can’t deliver within the deadline and the budget.

In the Travel industry requirements change frequently during the time the team is developing the software because market conditions are continuously changing. Creating a full requirement document is time consuming and not necessary since requirements often change.

Agile software development methods have recently emerged as a novel way of developing software compared with traditional methodologies. Agile software development aims at fast, light and effective development that supports the customers’ business without being chaotic or rigorous. Agile project management is defined as a conceptual framework for undertaking software engineering projects. Several approaches of Agile Methods are used to control and measure a software project. Agile Methods are considered a family of development processes, not a single approach to software development. The most popular and widely practiced Agile project management methods include Scrum, XP (Extreme Programming) etc.

This research study aimed to examine Agile project management in the travel industry and knowledge in different agile methodologies.

Keywords: Agile software development, agile methods, travel industry, Scrum, XP
Acknowledgments

I would like to thank my supervisor Yngve Lindsjørn for his encouragement, advice and guidance throughout this study.

I would like to thank the Tour Operator company and Software company for letting me perform the case study and also the Project Manager in the Tour Operator company for helping me get in touch with relevant people in the project and assisting me in finding the correct information from the project.

I would also like to thank my parents and my brothers for their infinite support and understanding in my life, and my sweet daughter Beren for missing out on some of her playtime and for her patience during this study.

Last, but not least, I would like to thank my wife, who provided everything I needed. This study would not have been possible without her love, support, encouragement and patience.
Contents

1 Introduction ............................................. 1
   1.1 Research Questions ................................. 3
   1.2 Target group ........................................ 3
   1.3 Personal motivation ................................. 3
   1.4 Research method in brief ......................... 4
      1.4.1 Literature study ............................... 4
      1.4.2 Case study ...................................... 4
      1.4.3 Survey ........................................... 4
      1.4.4 Observation ..................................... 4
   1.5 Structure of the report ............................. 5

2 Agile overview, definitions and characterizations .... 6
   2.1 What is Agile Project Management? ................. 6
   2.2 The Agile Manifesto ................................ 9
      2.2.1 Principles behind the Agile Manifesto ........ 10
   2.3 Agile software development methods: ............. 11
      2.3.1 Extreme programming ........................... 11
      2.3.2 Rational Unified Process ...................... 12
      2.3.3 Scrum ........................................... 13
      2.3.4 Feature Driven Development .................. 19
      2.3.5 Dynamic System Development Method .......... 19
      2.3.6 Adaptive Software Development ............... 20
      2.3.7 Kanban ........................................... 21
   2.4 Comparison of Agile Methods ...................... 21
   2.5 Disciplined agile delivery: ....................... 24
   2.6 Agile product testing ................................ 24
   2.7 Characteristics of successful Agile projects ..... 25
   2.8 Selecting an agile project ......................... 26

3 The travel and tourism industry ......................... 28
   3.1 Information technology in the travel industry ..... 28
   3.2 Stakeholders in the travel industry ............... 29
   3.3 The Tour Operator Company ........................ 29
      3.3.1 Need for Change ................................ 30
      3.3.2 Project purpose and description .............. 31
# 4 Methodology

4.1 Research Context .................................................. 34
4.2 Survey Methodology .............................................. 35
4.3 Case Study Methodology- Tour Operator Software .......... 36
4.4 Data Collection ..................................................... 37
4.5 Data Analysis ....................................................... 37

# 5 Research results

5.1 Scrum methods ....................................................... 40
  5.1.1 Product Backlog .............................................. 40
  5.1.2 Backlog Refinement Meeting .............................. 41
  5.1.3 Sprint Planning Meeting .................................. 43
  5.1.4 Sprint Backlog .............................................. 43
  5.1.5 Daily Scrum meeting ...................................... 45
  5.1.6 Sprint Review Meeting ................................... 46
  5.1.7 Sprint Retrospective Meeting ............................ 47

# 6 Discussion

6.1 Implications for Research ....................................... 49
6.2 Implications for Practice ....................................... 51
6.3 Answering the Research Questions ............................ 54
  6.3.1 Why do we need agile development in the travel industry? ............................................. 54
  6.3.2 What are the challenges in relation to the Agile methods? ........................................ 55
6.4 Recommendations .................................................. 59
6.5 Limitations .......................................................... 59
6.6 Future work ........................................................ 60

# 7 Conclusion

Appendices ............................................................... 66
A Interview guide .................................................... 67
Appendices ............................................................... 70
B Survey guide .......................................................... 71
List of Figures

1.1 Traditional vs Agile Software Development ................. 2
2.1 State of Agile survey sponsored by VersionOne ............ 7
2.2 State of Agile survey Benefits of Agile - sponsored by VersionOne .................................................. 8
2.3 Values, principles and practices of XP ..................... 12
2.4 Rational Unified process ................................... 13
2.5 Organization in Scrum .................................. 14
2.6 Disciplined Agile Delivery ............................. 24
2.7 Agile software development umbrella .................. 25
2.8 Agile Frame and software development ............ 27
3.1 Core Functionality .................................. 32
5.1 Qtest - Test Scenarios created by Tour Operator .... 39
5.2 Product Backlog - Version One .......................... 41
5.3 Splitting Stories .................................... 42
5.4 Velocity Trend - Version One ............................. 44
5.5 Sprint Backlog - Version One ............................ 45
B.1 Survey Q1 Result ................................... 72
B.2 Survey Q2 Result ................................... 73
B.3 Survey Q3 Result ................................... 74
B.4 Survey Q4 Result ................................... 75
B.5 Survey Q5 Result ................................... 76
B.6 Survey Q6 Result ................................... 77
B.7 Survey Q7 Result ................................... 78
B.8 Survey Q8 Result ................................... 79
B.9 Survey Q9 Result ................................... 80
List of Tables

1.1 Traditional Methodologies versus Agile Methodologies (Hany Wells & Smyth, 2015) ........................................... 2

2.1 General Features of Agile Methods. (Table modified from (M. Fowler, 2005)) ............................. 23

4.1 Role, years of experience and length of interview .................. 37
List of abbreviations

**SM:** Scrum Master

**PO:** Product Owner

**XP:** Extreme Programming

**TO:** Tour Operator

**RUP:** Rational Unified Process

**ASD:** Adaptive Software Development

**FDD:** Feature Driven Development

**DSDM:** Dynamic System Development Method

**DAD:** Disciplined agile delivery

**RAD:** Rapid Application Development

**PBI:** Product Backlog Items

**ITT:** Information Technology and Tourism

**SPO:** Special Price Offer
Preface
Chapter 1

Introduction

Both traditional and agile project delivery embody similar principles and practices that aim to deliver measurable results. Traditional project delivery can be described as a “waterfall” approach, which presumes that the requirements, expectations, duration, activities and outcomes of projects can be predicted accurately and planned in a sequence before any actual development activity takes place. With traditional project delivery the customer is not able to test the product until the end of the project when acceptance testing is performed and may, potentially, expose that the product is not delivering the expected functionality. As a result, the following factors are usually significant limitations of traditional project delivery in practice:

- Traditional Project Management focuses on plans and artifacts while Agile Project Management focuses on customer interaction and satisfaction. (“Agile Methodology,” 2016)
- In Traditional Project Management there is a response to change that is either corrective or preventative while in Agile Project Management changes are controlled through adaptive actions.
- Traditional Project Management is typically up-from planning while Agile Project Management is progressive elaboration with release and iterative planning.
- In Traditional Project Management there is top-down control while Agile Project Management has self-organizing and cross-functional teams.
- In Traditional Project Management, there is scope based delivery while Agile Project Management has time-boxed delivery.
- Traditional Project Management is contract oriented while Agile Project Management has customer orientation. …
### Traditional Methodologies versus Agile Methodologies

<table>
<thead>
<tr>
<th>Traditional Methodologies</th>
<th>Agile Methodologies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plan-driven prediction</td>
<td>Adaptive response to emergent change</td>
</tr>
<tr>
<td>Focus on planning the future in detail</td>
<td>Emphasis on adapting quickly to changing realities</td>
</tr>
<tr>
<td>Teams report on an exact planned set of actions</td>
<td>Teams change direction when the project changes</td>
</tr>
<tr>
<td>Inflexible division of projects into separate stages</td>
<td>Each stage’s tasks emerge from the outcome of the previous</td>
</tr>
<tr>
<td>Demand early detailed definition and commitments</td>
<td>Focus on workable functionality required to deliver business benefit</td>
</tr>
<tr>
<td>Depend on structure</td>
<td>Embrace creativity</td>
</tr>
<tr>
<td>Resist change</td>
<td>Welcome change</td>
</tr>
<tr>
<td>Slow to respond to requirement changes</td>
<td>Responding immediately to requirement changes</td>
</tr>
</tbody>
</table>

Table 1.1: Traditional Methodologies versus Agile Methodologies (Hany Wells & Smyth, 2015)

The Agile project delivery “is a way of managing projects to deliver customer value via adaptive planning, rapid feedback, continuous im-
provement and intense human interaction and collaboration”. Delivering “customer value” is a key aspect of agile project delivery. Agile project management is conducted through the collaboration of a small, co-located team that usually consists of the customer/end user, a project manager, a business analyst (or the role of business analysis) and specialist(s). Specialists could include system developers, subject matter experts, IT architect and/or the sole person with specific knowledge or expertise who understands how all the project pieces fit together. Agile project management uses an interactive process that helps customers define their needs and requirements. The Agile approach is suitable for complex projects where it is difficult to specify the product in advance. It is widely used in the software industry where the customer detects their needs through means of repeated tests and improvements to a prototype.

1.1 Research Questions

The research questions and the topic of this master thesis are as follows:

1. Why do we need Agile development in the travel industry?
2. What are the challenges in relation to the Agile methods?

1.2 Target group

The target group of this study is master students at the University of Oslo and other research institutions, as well as others interested in the field in general. Although I briefly define most of the ambiguous words in the paper, I take for granted that the reader has a basic understanding of concepts and topics in the field of Agile project management.

1.3 Personal motivation

This thesis contributes to academic achievement by enlightening an area where only limited formal research has previously been performed. The idea for the topic of this master thesis evolved from my interest in the travel business where I have been working as IT Manager. I manage the information technology department which implements and maintains policies and goals that support the Tour operator company’s IT needs.
1.4 **Research method in brief**

The main research methods used in this study are literature study, survey, observation and case study.

1.4.1 **Literature study**

This part of the research has been conducted by reviewing science articles and books. I have searched through an array of research documents dealing with the subjects addressed in this thesis. This study has helped to increase my understanding of the field and has provided the framework the study has been built upon.

The literature search within the field of innovation has provided me with a general understanding of different methods used in reviewing innovations. Additionally it has provided me with an understanding of agile project management in the travel industry.

1.4.2 **Case study**

The case study was performed in collaboration with the Tour Operator company in Scandinavia and the software provider that developed the back office system for Tour Operator.

The case study was divided into two: the first part was dedicated to gathering background information about Tour Operator, and the second part was dedicated to development of the Tour Operator back office system.

1.4.3 **Survey**

In order to collect necessary data an online survey was conducted. The goal of this survey was to attain information about what kind of knowledge the team had about agile and agile methodologies and whether they were satisfied with agile in the project.

1.4.4 **Observation**

Observation is great for investigating people’s behaviour - what they actually do, rather than what they say they do requires that you make sure that people have given their consent to being observed. I observed the agile team and joined some daily scrum meetings to gain a better understanding of agile project management.
1.5 Structure of the report

The rest of the paper is organized as follows: chapter two provides an overview of agile methodology and is followed by a chapter about the background of the travel industry. The fourth chapter addresses the research method including a detailed description of the context of the research, the choice of method, data collection, and data analysis. The results chapter present the findings from the study which are categorized into two sub sections. Finally, I state what I see as some of the main challenges and future directions for research on Agile project management in the travel industry.
Chapter 2

Agile overview, definitions and characterizations

This chapter provides a characterization and definition of agile software development, an overview of agile methodology and an overview of relevant research literature.

2.1 What is Agile Project Management?

Agile software development is an important topic in software engineering and information systems. (Dingsøyr, Dybå, & Moe, 2010). Agile project management is the process by which projects can be managed and implemented in small chunks of work. According to Williams and Cockburn (2003) agile software development is about feedback and change. The primary goal of all agile methods is to deliver software products quickly, and to adapt to changes in the process, product, environment, or other project contingencies (Aoyama, M., 1998). Agile projects deliver value to the business in frequent small deliveries of product called features. With traditional waterfall methodology the requirements for the project are documented up front. The the design of the whole solution is completed, followed by development, testing and finally, implementation of the product. If this whole process takes a year to complete, the business sees no tangible value until the very end of the project.

With agile projects, items are created by small logical chunks of work called iterations or sprints. Agile is a great technique to use when business needs are frequently changing or when the business wants to receive product benefits earlier. With Agile you can focus on what the business needs now and if that changes the new business needs can be accommodated in the next iteration. Agile is frequently used to manage IT projects, but can also be used to manage non-IT projects.

Examples of non-IT projects that are suitable for Agile are facility
moves, company reorganizations and, changing business processes within a department. Just about any project can utilize Agile if deliverables can be produced and implemented in a short period of time, and can be expanded or added to with future capabilities. Just like building blocks coming together, Agile projects build capabilities one piece or a few pieces at a time.

A survey sponsored by VersionOne. The 10th annual State of Agile survey, was conducted between July and November, 2015. There were 3,880 respondents to the survey, showing that agile methodology is frequently adopted and has many benefits. (“State of Agile Report,” 2015)

Figure 2.1: State of Agile survey sponsored by VersionOne
### Actual Improvements from Implementing Agile

The top three benefits of adopting agile have remained steady for the past five years: managing changing priorities (87%), team productivity (85%), and project visibility (84%).

<table>
<thead>
<tr>
<th>Benefit</th>
<th>Got Better</th>
<th>No Change</th>
<th>Got Worse</th>
<th>Don't Know</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ability to manage changing priorities</td>
<td>87%</td>
<td>3%</td>
<td>1%</td>
<td>9%</td>
</tr>
<tr>
<td>Increased team productivity</td>
<td>85%</td>
<td>3%</td>
<td>1%</td>
<td>11%</td>
</tr>
<tr>
<td>Improved project visibility</td>
<td>84%</td>
<td>3%</td>
<td>1%</td>
<td>12%</td>
</tr>
<tr>
<td>Increased team morale/motivation</td>
<td>81%</td>
<td>5%</td>
<td>3%</td>
<td>11%</td>
</tr>
<tr>
<td>Better delivery predictability</td>
<td>81%</td>
<td>6%</td>
<td>2%</td>
<td>11%</td>
</tr>
<tr>
<td>Faster time to market</td>
<td>80%</td>
<td>7%</td>
<td>1%</td>
<td>13%</td>
</tr>
<tr>
<td>Enhanced software quality</td>
<td>79%</td>
<td>6%</td>
<td>2%</td>
<td>14%</td>
</tr>
<tr>
<td>Reduced project risk</td>
<td>78%</td>
<td>6%</td>
<td>1%</td>
<td>15%</td>
</tr>
<tr>
<td>Improved business/IT alignment</td>
<td>77%</td>
<td>6%</td>
<td>1%</td>
<td>16%</td>
</tr>
<tr>
<td>Improved engineering discipline</td>
<td>73%</td>
<td>7%</td>
<td>2%</td>
<td>19%</td>
</tr>
<tr>
<td>Enhanced software maintainability</td>
<td>70%</td>
<td>8%</td>
<td>2%</td>
<td>21%</td>
</tr>
<tr>
<td>Better manage distributed teams</td>
<td>62%</td>
<td>11%</td>
<td>2%</td>
<td>25%</td>
</tr>
</tbody>
</table>

*Respondents were able to make multiple selections.

---

Figure 2.2: State of Agile survey Benefits of Agile - sponsored by VersionOne
2.2 The Agile Manifesto

Traditional project management emerged from the construction engineering and defense industries. Evolving from a meeting of major software development and IT industry leaders in 2001, who were concerned about creating better project management results for their clients, agile project management is a twenty-first century management approach. The 2001 meeting of experts resulted in the Agile Manifesto. The Agile Manifesto is written by a group of advocates of iterative and incremental development methods and is the foundation document of the agile movement. In combination with a set of 12 agile principles, it sets forth the underlying philosophical concepts of agile project management. It is important to note that agile project management encompasses all aspects of project delivery and not just the sole function of project management practices. It is inclusive of all business analysis, systems analysis and development, as well as all levels of quality assurance and testing. The manifesto is included here to enable the reader to understand that the “Go Agile” service’s objectives are to help the transition of an organization from its current projects delivery approach to one that is compatible with the principles outlined in the manifesto.

The principles of the Agile Manifesto are as follows:

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

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

*That is, while there is value in the items on the right, we value the items on the left more. (“Agile Manifesto,” 2001)*

Agile methodologies are not a single methodology; it is a group of software development methods based on iterative and incremental development approaches. In addition, agile methodology focuses on customer satisfaction and bug free products. According to Abrahamsson, P. et al. (2002), agile software development is characterized by four attributes i.e. it is incremental, cooperative, straightforward, and adaptive. Overall the methodology refers to software release with rapid developments, more collaboration and communication with customers, and changing requirements throughout the development cycle. It is more adaptive with the ability to react to the most recent changes. Agile methodology practices are friendlier towards the shifting priorities that are a fact of life in today’s fastpaced, competitive organizations and because of this, stakeholders are more satisfied with this process (Cohn, 2009). Agile methodology has resulted in faster growth and development in the software industry.
as well as improving the communication and collaboration both inside the agile team and with the customers. Extreme Programming (XP) and Scrum are the two dominant methods used in agile methodology. XP concentrates more on the development rather than managerial aspects of software projects, while Scrum method includes both Scrum management and Scrum development process.

2.2.1 Principles behind the Agile Manifesto

The 12 principles behind the Agile Manifesto are less known than the Manifesto but even more meaningful (Beck, 2001). These are:

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

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

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

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

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

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

7. Working software is the primary measure of progress.

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

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

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

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

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.
These principles are fairly simple in concept, but are profoundly deep in practice. Those principles have helped transform development practices and focus strongly on teamwork. The principles of Agile can be applied to any type of project or organization. "Agile is more a philosophy, a mindset, a way of thinking - a set of principles and values" (Peter Trudelle, 2013).

2.3 Agile software development methods:

2.3.1 Extreme programming

Extreme programming (XP) is a software development methodology which is intended to improve software quality and responsiveness to changing customer requirements. As a type of agile software development, it advocates frequent "releases" in short development cycles, by using sprints for one, two or three weeks, which are intended to improve productivity and introduce checkpoints at which new customer requirements can be adopted.

Other elements of XP include: programming in pairs or doing extensive code reviews, unit testing of all codes, avoiding programming of features until they are actually needed, a flat management structure, simplicity and clarity in codes, expecting changes in the customer’s requirements as time passes and the problem is better understood, and frequent communication with the customer and among programmers. The methodology takes its name from the idea that the beneficial elements of traditional software engineering practices are taken to "extreme" levels. As an example, code reviews are considered a beneficial practice; taken to the extreme, code can be reviewed continuously, i.e. the practice of pair programming.

Critics have noted several potential drawbacks, including problems with unstable requirements, no documented compromises of user conflicts, and a lack of overall design specification or documentation. The original XP recipe is based on four simple values, simplicity, communication, feedback and courage, and twelve supporting practices:

- Planning Game
- Small Releases
- Customer Acceptance Tests
- Simple Design
- Pair Programming
- Test-Driven Development
2.3.2 Rational Unified Process

Rational Unified Process (RUP) is an iterative software development process framework created by the Rational Software Corporation. RUP is not a single concrete prescriptive process, but rather an adaptable process framework, intended to be tailored by the development organizations and software project teams that will select the elements of the process that are appropriate for their needs. RUP is a specific implementation of the unified process. It is best suited for agile process.
2.3.3 Scrum

"Scrum is a framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value". ("Scrum Guide," 2013)

"Scrum is an iterative and incremental agile software development methodology for managing product development. It defines "a flexible, holistic product development strategy where a development team works as a unit to reach a common goal", challenges assumptions of the "traditional, sequential approach" to product development, and enables teams to self-organize by encouraging physical co-location or close online collaboration of all team members, as well as daily face-to-face communication among all team members and disciplines in the project." (Wikipedia, 2016)

A key principle of scrum is its recognition that during a project the customers can change their minds about what they want and need (often called "requirements churn"), and that unpredicted challenges cannot be easily addressed in a traditional predictive or planned manner. As such, scrum adopts an empirical approach—accepting that the problem cannot be fully understood or defined, focusing instead on maximizing the team’s ability to deliver quickly and respond to emerging requirements.
Brief review of the Scrum framework

Sprint planning for product development:

A sprint (or iteration) is the basic unit of development in scrum. The sprint is time boxed effort; that is, it is restricted to a specific duration. The duration is fixed in advance for each sprint and is normally between one week and one month, with two weeks being the most common.

Each sprint starts with a sprint planning event, the aim of which is to define a sprint backlog, where the work for the sprint is identified and an estimated commitment for the sprint goal is made. Each sprint ends with a sprint review and a sprint retrospective, where the progress is reviewed, shown to the stakeholders and areas for improvement for subsequent sprints are identified.

Scrum emphasizes the working product at the end of the sprint that is really done; in the case of software, this would probably include that the software has been integrated, fully tested, end-user documented, and is potentially shippable.
Sprint-planning event:

Sprint planning is the first event inside a Sprint. The Scrum Team plans the items they are going to deliver in the Sprint and the way they will deliver them.

- Select the work to be done.
- Prepare the sprint backlog i.e details concerning the time it will take to do the work, with the entire team.
- Identify and communicate how much of the work is likely to be done during the sprint.
- Four-hour time limit for a two-week sprint (pro rata for other sprint durations).
- During the first half, the whole team (development team, scrum master, and product owner) agree what product backlog items they will consider for the sprint.
- During the second half, the development team establish the work (tasks) required to deliver those backlog items, resulting in the sprint backlog.

Daily scrum:

Each day during a sprint, the team hold a daily scrum (or stand-up) with specific guidelines:

- All members of the development team come prepared.
- The daily scrum starts precisely even if some development team members are missing.
- The daily scrum should happen at the same location and same time every day.
- The daily scrum length is limited (time-boxed) to fifteen minutes.
- Anyone is welcome, although normally only the scrum team roles contribute.

During the daily scrum, each team member answers three questions:

- What did I do yesterday that helped the development team achieve the sprint goal?
- What will I do today to help the development team achieve the sprint goal?
• Do I see any impediment that prevents me or the development team from achieving the sprint goal?

Any impediment (stumbling block, risk, or issue) identified in the daily scrum should be captured by the scrum master, displayed on the team’s scrum board, and someone should take the responsibility for working toward a resolution outside of the daily scrum. There should be no detailed discussions during the daily scrum.

**Sprint review and sprint retrospective:**

At the end of a sprint, the team holds two events: the sprint review and the sprint retrospective. **At the sprint review the team:**

• Reviews the work that was completed and the planned work that was not completed.
• Presents the completed work to the stakeholders.

**Guidelines for sprint reviews:**

• Incomplete work cannot be demonstrated.
• The recommended duration is two hours for a two-week sprint (pro-rata for other sprint durations).

**At the sprint retrospective, the team:**

• Reflects on the past sprint.
• Identifies and agrees continuous process improvement actions.

**Guidelines for sprint retrospectives:**

• Two main questions are asked in the sprint retrospective: What went well during the sprint? What could be improved in the next sprint?
• The recommended duration is one-and-a-half hours for a two-week sprint (pro-rata for other sprint durations).
• This event is facilitated by the scrum master.
Product backlog:

The product backlog comprises an ordered list of requirements that a scrum team maintains for a product. It consists of features, bug fixes, non-functional requirements, etc.—whatever needs doing in order to successfully deliver a viable product. The product owner orders the product backlog items (PBIs) based on considerations such as risk, business value, dependencies, and date needed. Items added to a backlog are commonly written in story format. The product backlog is what will be delivered, ordered into the sequence in which it should be delivered. It is open and editable by anyone, but the product owner is ultimately responsible for ordering the items on the backlog for the development team to choose.

The product backlog contains the product owner’s assessment of business value and the development team’s assessment of development effort, which are often, but not always, stated in story points using a rounded Fibonacci sequence. These estimates help the product owner to gauge the timeline and may influence ordering of backlog items; for example, if the "add spellcheck" and "add table support" features have the same business value, the product owner may schedule earlier delivery of the one with the lower development effort (because the return on investment is higher), or the one with higher development effort (because it is more complex or risky, and they want to eliminate that risk earlier).

The product backlog and the business value of each backlog item is the responsibility of the product owner. The size (i.e. estimated complexity or effort) of each backlog item is, however, determined by the development team, who contributes by sizing items, either in story points or in estimated hours.

There is a common misunderstanding that only user stories are allowed in a product backlog. By contrast, scrum is neutral on requirement techniques. As the Scrum Primer states: "Product Backlog items are articulated in any way that is clear and sustainable. Contrary to popular misunderstanding, the Product Backlog does not contain "user stories"; it simply contains items. Those items can be expressed as user stories, use cases, or any other requirements approach that the group finds useful. But whatever the approach, most items should focus on delivering value to customers." (Pete Deemer & Vodde, 2012)

Scrum advocates that the role of the product owner is assigned. The product owner is responsible for maximizing the value of the product. The product owner gathers input and takes feedback from and is lobbied by, many people, but ultimately makes the call on what gets built.
The product backlog is used to capture requests for modifying a product. This can include adding new features, replacing old features, removing features and fixing issues to ensure the development team is given work which maximizes the business benefit to the owner of the product.

Typically, the product owner and the scrum team come together and write down everything that needs to be prioritized and this becomes the content of the first sprint, which is a block of time meant for focused work on selected items that can be accommodated within a time frame. The product backlog can evolve as new information surfaces about the product and about its customers, and subsequent sprints may address new work.

The following items typically comprise a scrum backlog: features, bugs, technical work, and knowledge acquisition. Web development can entail confusion as to the difference between a feature and a bug: technically a feature is "wanted", while a bug is a feature that is "unintended" or "unwanted" (but may not necessarily be a defective thing). An example of technical work would be: "run virus check on all developers’ workstations". An example of knowledge acquisition could be a scrum backlog item about researching Wordpress plugin libraries and making a selection.

Managing the product backlog between product owner and scrum team:
A backlog, in its simplest form, is merely a list of items to be worked on. Having well-established rules about how work is added, removed and ordered helps the whole team make better decisions about how to change the product.

The product owner prioritizes which of the items in the product backlog are most needed. The team then chooses which items they can complete in the coming sprint. On the scrum board, the team moves items from the product backlog to the sprint backlog, which is the list of items they will build. Conceptually, it is ideal for the team to only select what they think they can accomplish from the top of the list, but in practice it is not unusual to see that teams are able to take lower-priority items from the list along with the top ones selected. This normally happens because there is time left within the sprint to accommodate more work. Items at the top of the backlog, the items that are going to be worked on first, should be broken down into stories that are suitable for the development team to work on. The further down the backlog goes, the less refined the items should be. As Schwaber and Beedle put it "The lower the priority, the less detail, until you can barely make out the backlog item." (Schwaber & Beedle, 2002)

As the team works through the backlog, it needs to be assumed that "changes in the world can happen"—the team can learn about new market opportunities to take advantage of, competitor threats that arise, and feedback from customers that can change the way the product was meant to
work. All of these new ideas tend to trigger the team to adapt the backlog to incorporate new knowledge. This is part of the fundamental mindsets of an agile team; the world changes, the backlog is never finished.

2.3.4 Feature Driven Development

The feature driven development (FDD) variant of agile methodology was originally developed and articulated by Jeff De Luca, with contributions by M.A. Rajashima, Lim Bak Wee, Paul Szego, Jon Kern and Stephen Palmer. FDD is a model-driven, short-iteration process. It begins with establishing an overall model shape. Then it continues with a series of two-week “design by feature, build by feature” iterations. The features are small, “useful in the eyes of the client” results. FDD designs the rest of the development process around feature delivery using the following eight practices:

- Domain Object Modeling
- Developing by Feature
- Component/Class Ownership
- Feature Teams
- Inspections
- Configuration Management
- Regular Builds
- Visibility of progress and results

FDD recommends specific programmer practices such as “Regular Builds” and “Component/Class Ownership”. FDD’s proponents claim that it scales more straightforwardly than other approaches, and is better suited to larger teams. Unlike other agile methods, FDD describes specific, very short phases of work, which are to be accomplished separately per feature. These include Domain Walkthrough, Design, Design Inspection, Code, Code Inspection, and Promote to Build. (McLaughlin, 2015)

2.3.5 Dynamic System Development Method

Dynamic system development method (DSDM), dating back to 1994, grew out of the need to provide an industry standard project delivery framework for what was referred to as Rapid Application Development (RAD) at the time. While RAD was extremely popular in the early 1990’s, the RAD approach to software delivery evolved in a fairly
unstructured manner. As a result, the DSDM Consortium was created and convened in 1994 with the goal of devising and promoting a common industry framework for rapid software delivery. Since 1994, the DSDM methodology has evolved and matured to provide a comprehensive foundation for planning, managing, executing, and scaling agile processes and iterative software development projects.

DSDM is based on nine key principles that primarily revolve around business needs/values, active user involvement, empowered teams, frequent delivery, integrated testing, and stakeholder collaboration. DSDM specifically states “fitness for business purpose” as the primary criteria for delivery and acceptance of a system, focusing on the useful 80 percent of the system that can be deployed in 20 percent of the time.

Requirements are baselined at a high level early in the project. Rework is built into the process, and all development changes must be reversible. Requirements are planned and delivered in short, fixed-length time-boxes, also referred to as iterations, and requirements for DSDM projects are prioritized using MoSCoW Rules:

- M - Must have requirements
- S - Should have if at all possible
- C - Could have but not critical
- W - Won’t have this time, but potentially later

All critical work must be completed in a DSDM project. It is also important that not every requirement in a project or time-box is considered critical. Within each time-box, less critical items are included so that if necessary, they can be removed to prevent them impacting higher priority requirements on the schedule. The DSDM project framework is independent of, and can be implemented in conjunction with, other iterative methodologies such as XP and the RUP. (McLaughlin, 2015)

2.3.6 Adaptive Software Development

Adaptive software development (ASD) provides a framework for the iterative development of large, complex systems. The method encourages incremental, iterative development with constant prototyping (Highsmith, 1996). ASD highlights that a sequential waterfall approach only works in well understood and well-defined environments. But as changes occur frequently in software development, it is important to use a change-tolerant method. The first cycles of an ASD project should be short, ensure that the customer is strongly involved and confirm the project’s viability. Each cycle ends with a customer focus group review. During the review meetings a working application is explored.
2.3.7 Kanban

Kanban is a method for managing knowledge work with an emphasis on just-in-time delivery while not overloading the team members. In this approach, the process, from definition of a task to its delivery to the customer, is displayed for participants to see. Team members pull work from a queue.

Kanban in the context of software development can mean a visual process-management system that tells what to produce, when to produce it, and how much to produce.

Start with existing process:
The Kanban method does not prescribe a specific set of roles or process steps. The Kanban method starts with existing roles and processes and stimulates continuous, incremental and evolutionary changes to the system. The Kanban method is a change management method.

Agree to pursue incremental, evolutionary change:
The organization (or team) must agree that continuous, incremental and evolutionary change is the way to make system improvements and make them stick. Sweeping changes may seem more effective but have a higher failure rate due to resistance and fear in the organization. The Kanban method encourages continuous small incremental and evolutionary changes to the current system.

Respect the current process, roles, responsibilities and titles:
It is likely that the organization currently has some elements that work acceptably and are worth preserving. The Kanban method seeks to eliminate fear in order to facilitate future change. It attempts to eliminate initial fears by agreeing to respect current roles, responsibilities and job titles with the goal of gaining broader support.

Leadership at all levels:
Acts of leadership at all levels in the organization, from individual contributors to senior management, are encouraged.

2.4 Comparison of Agile Methods

All agile methods consider agile principles but they have different ways to follow these principles. Let me provide an example in the travel industry. All airline and charter operators’ main goal is to fill their planes and earn
money but they follow different strategies and do different marketing to reach their main goal. Agile methodologies do the same. They have different principles to reach their main goal. The main goal is to be agile.

Comparison often implies evaluating one method against the other. Table 2.1 below discusses each method using three selected aspects: key points, special features and identified weakness. Key points detail the methods, principles, aspects or solution. Special features describes one or several aspects of the methods that differentiate them from others. Finally, identified weakness relates to some aspects of a method that have been documented in the literature.

XP advocates test driven development where tests are used to specify the behavior of a program before the program code is actually written. Scrum is a project management approach that relies on self-organizing independent teams implementing a software project in 30-day cycles called sprints. Scrum is designed for a wide range of software projects of various sizes and can also be adapted to non-software projects such as product/application development. There are some differences between scrum and XP. One of the differences is XP teams work in iterations that are one or two weeks while scrum teams work in sprints that are two weeks to a month. ASD is designed for project environments that are high-speed, high-change, and high-uncertainty and is also good for organizations that are highly adaptive. FDD is designed, and primarily used for large, complex banking projects and is good for any large, complex project with large teams. DSDM is designed for any technical or business environment where a system needs to be quickly developed and deployed; preferably any project that is a good match the DSDM method as outlined in the Project Approach Questionnaire.
<table>
<thead>
<tr>
<th>Method Name</th>
<th>Key Points</th>
<th>Special features</th>
<th>Identified weakness</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASD</td>
<td>Adaptive culture, collaboration, mission-driven component based iterative development</td>
<td>Organizations are seen as adaptive systems. Creating an emergent order out of a web of interconnected individuals</td>
<td>ASD is more about concepts and culture than the software practice.</td>
</tr>
<tr>
<td>DSDM</td>
<td>Application of controls to RAD, use of time-boxing and empowered DSDM teams.</td>
<td>First truly agile software development method, use of prototyping, several user roles: “ambassador”, “visionary” and “advisor”</td>
<td>While the method is available, only consortium members have access to white papers dealing with the actual use of the method.</td>
</tr>
<tr>
<td>XP</td>
<td>Customer driven development, small teams, daily builds</td>
<td>Re-factoring - the ongoing redesign of the system to improve its performance and responsiveness too change</td>
<td>While individual practices are suitable for many situations, overall view and management practices are given less attention.</td>
</tr>
<tr>
<td>SCRUM</td>
<td>Independent, small, self organizing development teams, 30-day release cycles.</td>
<td>Enforce a paradigm shift from the “defined and repeatable” to the “new product development view of Scrum”</td>
<td>While Scrum details in specific how to manage the 30-day release cycle, the integration and acceptance tests are not detailed.</td>
</tr>
<tr>
<td>FDD</td>
<td>Five-step process, object oriented component (i.e. feature) based development.</td>
<td>Method simplicity, design and implement the system by features, object modeling.</td>
<td>FDD focuses only on design and implementation. Needs other supporting approaches.</td>
</tr>
</tbody>
</table>

Table 2.1: General Features of Agile Methods. (Table modified from (M. Fowler, 2005))

23
2.5 Disciplined agile delivery:

Disciplined agile delivery (DAD) is a process decision framework that enables simplified process decisions around incremental and iterative solution delivery. DAD builds on the many practices espoused by advocates of agile software development, including Scrum, agile modeling, lean software development, and others.

The primary reference for disciplined agile delivery is the book of the same name, written by Scott Ambler and Mark Lines.

In particular, DAD has been identified as a means of moving beyond Scrum. “The DAD framework provides a carefully constructed mechanism that not only streamlines IT work, but more importantly, enables scaling.” Paul Gorans and Philippe Kruchten call for more discipline in implementation of agile approaches and indicate that DAD, as an example framework, is “a hybrid agile approach to enterprise IT solution delivery that provides a solid foundation from which to scale. (Kruchten & Gorans, 2014)

2.6 Agile product testing

Agile product testing is a software testing practice that follows the principles of agile software development. Agile testing involves all members of a cross-functional agile team, with special expertise contributed by testers, to ensure deliver of the business value desired by the customer at frequent intervals, working at a sustainable pace. Specification by example is used to capture examples of desired and undesired behaviour and guide coding.

Agile development recognizes that testing is not a separate phase, but an integral part of software development, along with coding.
teams use a "whole-team" approach to "baking quality in" to the software product. Testers on agile teams lend their expertise by eliciting examples of desired behaviour from customers, collaborating with the development team to turn those into executable specifications that guide coding. Agile testing covers all types of testing. The Agile Testing Quadrants provide a helpful taxonomy to help teams identify and plan the testing needed.

2.7 Characteristics of successful Agile projects

Successful Agile project have particular characteristics: the sprints or iterations are typically four weeks long, face to face communication is prioritized above documentation (we want to produce a product, not product documentation), business and technical team members are collocated or use very rich virtual tools to simulate being together, and there is a sponsor who is 100% committed to the Agile process. Finally, requirement changes are anticipated and accommodated. There are other characteristics required for Agile projects to work that do not differ from traditional projects. These include having a vision for the end game, following a universally understood project lifecycle, understanding the requirements, the use of a shared and managed schedule with a dedicated team that is focused on getting the job done. Lastly, communication with all stakeholders is critical. There are several agile methodologies and processes in use today.
2.8 Selecting an agile project

One of the most important success factors for agile is the selection of the right project to apply agile methods. Good agile project candidates share these characteristics. First, they need a quality product delivered quickly but not all at once. Secondly, they expect the requirements to change or evolve. In addition, the organization is willing to release very capable team members that can independently make decisions about the product that is being produced, and finally, the product can deliver business value in incremental pieces.

Let’s look at a few examples of agile and non-agile projects to put this in perspective. Let’s say you want to implement an executive information system. You have an agreed budget, strong executive buy-in, and you have seven executives, each with very different needs. They definitely want something delivered sooner rather than later, and they want the project done by the end of the year. This project has agile written all over it. The requirements will undoubtedly change and evolve as the executives start to see the possibilities. You can easily break down the project into sprints, as you could categorize and implement features.

Let’s look at another example. You need to do a business process refresh, replacing administrative processes with updated ones to meet greater customer demand. This is something your company does every three to four years, with a proven set of steps and processes to follow. The budget is agreed, all the scope must be completed, and you know what business areas and the number of processes that must be updated. This is not a good candidate for agile. It does not serve a purpose to perform sprints and ask if there are new features required at the end of each sprint. The project’s been completed in the past using other methods which have worked, and people are used to. Could this project be implemented in sprint-like iterations? Maybe, but just doing something in iterations does not make it agile. We know all the processes must be refreshed so it’s not okay to time-box the project, and only allow for some of the processes to be refreshed based on a pre-agreed schedule, which could happen with an agile project. In this case it is better to use the more traditional approach which has worked in the past. Not all projects have to be implemented as just an agile, or non-agile project. It is possible to implement a project as a hybrid, of agile and non-agile. Take an introduction in a new product line in an existing business, for example. I might use agile techniques to develop new marketing products and get as much as possible done by the desired product delivery date. I would use a more traditional life cycle for the reorganization of people into the new product department, as each person must be evaluated, have their job responsibilities changed as required, and have their new job created. I would apply the changes to
personnel first, and then the business process changes could be made by the newly reassigned staff. Using agile for the process changes is wise, as it forces the company to carefully prioritize the work, and implement the most important features and business process changes first. Putting people into new positions is best done in a more traditional project approach.

In summary, agile can be applied to several different scenarios and be implemented in different ways. It all depends on the project and organization you’re working in, and whether the characteristics of agile are the best way to create change for customers.

![Agile Framework](image)

Figure 2.8: Agile Frame and software development
Chapter 3

The travel and tourism industry

The travel and tourism industry is one of the biggest and fastest growing industries in the world. The increasing availability and flexibility of many forms of technology offers a range of opportunities and challenges in many sectors including tourism. Technology is frequently described as a key driver for increasing the mobility of tourists, and therefore, for sustaining the growth of the tourism industry (Laws E. & G. Moscardo, 1998) (Bacchus & Molina, 2001).

There are two words that best characterize the future for the travel industry and these are growth and change. Many global forces have driven companies in this industry to adapt and change quickly to survive and remain competitive (Tjoa, 1997).

Technology has played an important role in achieving economic growth by garnering economic means for consumers to travel. The development and contribution of air travel as a means of transporting tourists to different locations is an important contribution of technology to the tourism and travel industry. The significant development of low cost carriers is another example of successful technological application to the tourism and travel industry (Stipanuk, 1993).

The tourism industry is often defined as the sectored system of innovation and production. This sectored system in tourism is extremely complex, given the fact that the tourism products and participants change due to changes in external forces.

3.1 Information technology in the travel industry

Information Technology and Tourism (ITT) is the first scientific interdisciplinary journal focusing on the nature and role of information technology within the context of tourism, travel and hospitality. Information and communication systems embedded in a global net have had a profound influ-
ence on these industries, while these industries, with their presence in the electronic market, have an impact on the developments of IT. Advances in the use and development of tools, technologies, and methodologies have facilitated the efficient netting of information and communication systems in tourism (Tracy Ireland and John Schofield, 2015).

3.2 Stakeholders in the travel industry

The different stakeholders in the travel industry would be the sales team, the marketing team and the customers. Their input should be acquired and compiled appropriately. The sales team would be required to demonstrate the product properly and collect the inputs for product improvements. Likewise the marketing team would be required to follow the same strategy. The project / product manager team should give a proper demonstration to the customer and collect their feedback regarding the product.

In the travel industry customers are unable to definitively state their needs up front. The next section provides details about a tour operator company in Scandinavia and their software needs.

3.3 The Tour Operator Company

The Tour Operator company, are the specialist tour operators to Turkey, Egypt, Spain, Greece and Thailand. The company is one of the most reliable operators in the Scandinavian market for genuine sunny holidays and traveling in style. They offer comfort, courtesy and individuality as well as exceptional opportunities to ensure that their guests experience joyful and memorable holidays, and their reputation means great value.

Corporations that operate in international markets are faced with an increasingly dynamic, changeable and unpredictable world, where they need to control their activities in order to be efficient and productive. My case has proved to fit this description; it spans over multiple countries whose rules it must adhere to, it provides services to diverse customers, and it employs different suppliers and cooperates with various external systems and services.
3.3.1 Need for Change

The need for change in the system arose from:

- Increasing competition between other tour operators with more client-friendly solutions.

- The fact that the old booking system was becoming outdated. It did not cover the needs of the employees and external agents, and could not be expanded to do so. They were thus forced to do a lot of unnecessary work manually and The Tour Operator ended up hiring unnecessary staff for the operation departments. Particularly regarding external agents, the limitations in functionality could result in fewer using The Tour Operator’s services with consequential losses.

- Working with outdated technology was a problem for the software provider. It was hard for them to find a workforce willing to further develop an already ‘dying’ product. As time passed, The Tour Operator was increasing their sales, but competition was also increasing. They noticed that their vendor was not improving in terms of technology, support and documentation. This was seen as a potential risk, so they were looking for alternatives.

- The booking system not being integrated with the website, something that aggravated the aforementioned problems.

- The booking system not being centralized. Some of the components had to be installed separately in each sub-location and the databases of each international office did not communicate.

- Negative feedback from users.

Hence, it was imperative to replace some of the unsuitable components, and reform the rest. Before applying any changes, a survey was conducted among the clients and the external agents in order to define and consider their needs.

It should firstly have a supporting or enabling function. Both the technical components supporting The Tour Operator’s operations, and the connections between the company and their diverse partners, automate and improve existing activities. However, they also open up a field of activities that were not possible before thereby increasing the company’s profit. The interviewees mention that after the changing of the system, new ideas for functionality surfaced. Such was the case with the generation of reports, sending SMSs, or the ability of third parties to connect with the Tour Operator. In order to be considered an Information Infrastructure, a system should be shared by a larger community. So
far I have stated that clients, personnel and third parties interact with the technological components, each one through a different interface. The same objects appear differently to each of them, and the levels of access vary. The system should also be open. There are indeed no limits posed to the number of customers and external agents. Of course, due to the nature of such commercial organizations, groups of users like the employees remain limited, and so do stakeholders such as banks. Similarly, the application areas are restricted but expansions in technological components and nodes in this heterogeneous network are applicable and allow evolution. When a system acquires the status of a heterogeneous network, several activities occur, relations vary and requirements may change, and propagate even more changes.

3.3.2 Project purpose and description

The purpose of this project is as follows:
Tour Operator’s current software was implemented in 2010 and senior level management determined that it was not effective in providing a thorough description of their services. Additionally, the system was not efficient or user friendly.

Project Description:
The Tour Operator company was seeking a provider to utilize the latest technology to implement a new tour operator system that:

- Provided detailed information about their services
- Provided web services and APIs
- Met the expectations of business requirements
- Qualified management system
- Was powerful
- Was flexible
- Could easily transfer existing data to new software
- Could easily add or modify features and handle their future tasks
- Was user friendly
3.3.3 Project Scope

The Tour Operator company has number of offices in Scandinavia and all markets work independently so they expect to have software solutions where most of features are optional allowing each market to configure the features according to their needs. The system must allow the user to generate, edit, delete, and archive information concerning flights, hotels, transfers and excursions enabling the customer to create his/her holiday package. When creating the holiday package, the system must quickly and accurately calculate the prices according to the user input. The customer should be able to make and check the reservation, change or delete services and passengers, pay, print/send invoices to the other passengers, and save various reports.

Some of the most important features that Tour Operator expects to have in their software are:

- Definition of bank, currency exchange rate, location, airlines, agency, agency commission, supplement, discount, profit margin rules, reservation and cancellation rules, hotels, flights, insurance, excursions, transfer, rooms and accommodation, boards, hotel/room facilities, credit card, cancellation reasons, invoice templates, grouping of
hotels, promotions, hotel/flight stop sale, release for hotel room and
service extras, hotel/flight optimization, special price offers (SPO)

- Automatic price calculation.
- Dynamic packaging
- Tracking payments
- Task Manager
- Issuing documents
- Reservation search and reservation form
- Package and ticket price search wizard
- Reports
- Incoming
- User rights
Chapter 4

Methodology

This chapter concerns the methodology. I conducted a case study to acquire a better understanding of why agile development is needed in the travel industry and what the challenges are.

The first section describes the context of the research. The second and third section outlines and justifies the choice of method. The fourth describes how the data collection was conducted and the fifth section describes how data analysis was performed.

4.1 Research Context

A software company with experience in the travel industry was selected as the focus for my research. The company will be referred to as “IT company” since they preferred to remain anonymous due to the company’s business policy.

IT company is considered an energetic and youthful organization that is responsible for providing advanced tour operator software (incoming/outgoing), dynamic packages, mobile application, CRM, B2C, CMS, and B2B. The company aims to become the main provider of usable, innovative and engaging social and transactional real estate services through knowledge, integrity and skilled execution. The company consists of experienced employees who have worked in the field for the last 10 years. They are highly qualified professionals who are able to meet the requirements. IT company make various web based solutions, converting existing technology projects into advanced technologies. The company’s financial situation is acceptable.

The company uses agile methodology, specifically the semi scrum method to manage their overall work. The company conducts long term, complex projects and used to apply traditional waterfall methodology to software development projects, but the last year they have adopted Scrum for this purpose.
The last project they conducted for was for Tour Operators. When considering which methodology to use they considered the following factors:

- Project size: large size and complex
- Customer Availability: frequently available through the duration of the project
- Level of integration with external systems: simple
- Customer tolerance for scope and cost changes: fixed budget but tolerate and welcome changes to the cost and schedule.
- Time to market: rapid deployment needed

Having defined and analyzed requirements they concluded that Agile methodology was suitable for the project and it is this software project I have studied.

The project team comprised eight people: one product owner, one scrum master and six in the development team.

The Tour Operator composed a list of requirements that needed to be fulfilled in the new system. The IT company surveyed other systems and made suggestions regarding the functionality to be implemented. They also interviewed The Tour Operator’s sub-companies in Scandinavia to get a better grasp of the requirements. During the development of the new system, The Tour Operator was kept updated through meetings. Four people from The Tour Operator worked together with IT company: two of them sat in the same room as IT company’s team and two of them visited to IT company’s team regularly.

### 4.2 Survey Methodology

In order to collect necessary data an online survey was conducted. This survey was performed in April 2016 and there were 9 respondents. The survey was sent out to the Scrum team involved in the Tour Operator software project. The goal of this survey was to attain information about what kind of knowledge the team had about agile and agile methodologies and whether they were satisfied with agile in the project. The survey result may be seen in Appendix B.
4.3 Case Study Methodology- Tour Operator Software

"A case study is an empirical inquiry that investigates a contemporary phenomenon in depth and within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident. In other words you use the case study method because you want to understand a real-life phenomenon in depth, but such understanding encompasses important contextual conditions – because they were highly pertinent to your phenomenon of study" (Yin, 2009)

Case studies are considered useful in research as they enable researchers to examine data at the micro level.

There are several reasons for why I chose a case study to investigate my problem. Case studies are, according to Yin (2009), preferred when the research question asks “how” or “why”. Because one of my problem formulation starts with “why”, a case study seemed like a good choice. The use of case studies is also preferred when the researcher has little or no control over the events that are studied and when the focus, according to Yin (2009), is on a contemporary phenomenon in a real-life context.

Consequently, case study methods are, often viewed as a support discipline, at best able to deliver ideas that can be tested in quantitative studies.

I conducted a case study to acquire a better understanding of how one can, and should, perform software development in a real agile project. The project was large and complex and there were several stakeholders. The system was developed for the parent company but four sub-companies were also going to use it. Almost all the system functionality requirements were the same but some of sub-companies did have different requests.

This case study aimed to investigate a tour operator’s history and their requirements for a back office system.

I based my study on interviews conducted with the various participants in the project. The project manager of Tour Operator gave me material and information to aid my understanding of the project and the processes involved.
4.4 Data Collection

An interview guide was prepared for data collection. The interviews were semi-structured and a short demonstration of the booking system was also made. The interview questions may be seen in Appendix A. A few structured questions were also included in case some topics of interest were not addressed by the initial questions. The interviewees familiarized me with the history of the company and stressed that they adopt a client-oriented philosophy. They specified who the users of the booking system are, the reasons why a shift was demanded, the functional and nonfunctional requirements of the system that would address their demands, and how these requirements were elicited. They talked about the providers that develop the components, the technologies used, and the way these components are connected. They described the development procedure of the new system and the training process the employees must undergo. Finally, they evaluated the outcomes of introducing the new system.

<table>
<thead>
<tr>
<th>Role in the project</th>
<th>Years of experience in using Agile/Scrum method</th>
<th>Length of interview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Manager/Product Owner</td>
<td>2 years</td>
<td>1:05:05</td>
</tr>
<tr>
<td>Scrum Master/ Project Manager</td>
<td>2 years</td>
<td>0:48:50</td>
</tr>
<tr>
<td>Senior Developer</td>
<td>5 years</td>
<td>0:45:37</td>
</tr>
<tr>
<td>Tester/Trainee</td>
<td>2 years</td>
<td>0:42:50</td>
</tr>
<tr>
<td>Project Manager from Tour Operator</td>
<td>10 years</td>
<td>1:25:42</td>
</tr>
</tbody>
</table>

Table 4.1: Role, years of experience and length of interview

All interviews were conducted in person and recorded. Notes were also made during each interview and the recordings acted as support in the data analysis of the material gathered from the interviews. The interviews were anonymous in order to allow the interviewees to speak freely.

4.5 Data Analysis

The principle data analysis method for case studies is referred to as OTTR, which stands for “observe,” “think,” “test,” and “revise.” Analysis must be an iterative process whereby the initial observations are reflected upon and shape subsequent data collection. According to Hartley (1994), data collection and analysis are “developed together in an iterative process,” which can be a strength as it allows for theory development which is
grounded in empirical evidence. Besides, a careful description of the data and the development of categories in which to place behaviors or processes have proven to be important steps in the process of analyzing the data. The data may then be organized around certain topics, key themes or central questions, and finally the data need to be examined to see how far they fit or fail to fit the expected categories.

Kohlbacher (2006) addressed qualitative content analysis in case study research. He indicated that “experts in research recommend conducting interpretations of results on two levels: interpretation of the results of one’s own survey and comparative interpretation of results and conclusions of existing theories and research results” (Kohlbacher, 2006). I have used openness and grounded analysis of the empirical data with a focus on identifying Agile project management in the travel industry. Grounded theory can help gather rich data through data collection and help get started with early data analysis (Glaser & A.Strauss, 1967). In order to acquire an overview of the challenges the Agile project management, I have conducted a literature review.

The research study was conducted through face to face interviews with four members from IT-company. A grounded analysis was chosen for analysis of the transcribed data and all the challenges experienced by the respondents are listed. Furthermore, I performed comparative interpretation of explicitly stated comments in the transcribed data and the literature review. As a result of the analysis I was able to identify some general challenges in relation to the Scrum method from the empirical data. However, I noticed that there were contradiction in the participants’ response. Therefore, I used an open- minded approach; a holistic and comprehensive approach towards data analysis, in an attempt to grasp the challenges and issues in the data as a whole rather than by analyzing line by line. Hence, this approach helped find some new issues which have not previously been reported.

Analysis of the data exposed some general challenges and some new issues. The thesis produces the issues and challenges facing the company in relation to applying Agile methodology. The categorized challenges and issues are detailed in results and discussion section.
Chapter 5

Research results

This chapter presents the research results. On the basis of data analysis, some general challenges experienced by the respondents from IT company and Tour Operator are listed.

The Tour operator company had already prepared a list of requirements to obtain offers for this project from various IT companies. They used Qtest tools and created requirements, test scenarios etc. However, the list was not very detailed because they thought that more details could be provided during project development.

![Figure 5.1: Qtest - Test Scenarios created by Tour Operator](image)

One of the hardest tasks in software development is to determine how long and how much work is required to deliver a new software product. Customers always want to know the cost of the project and when it will be delivered. Estimating a software project is never an easy task, and the situation is even worse when you have to do it without first understanding the context or the requirements well enough. To give an estimated date and offer to IT companies, you must have several meetings with some of the key employees to go through the requirements and gain a better understanding of the scope of the project.
The Tour operator was flexible with regard to delivery time but they had a limited budget and wanted to pay a fixed price. Even though they were flexible with regard to delivery time, they wanted to know when the project could be delivered and they agreed that IT Company would work with some key persons from Tour Operator Company. The entire team and persons from Tour Operator would sit in the same room to promote quick feedback in the case of unclarities.

The Scrum methods that IT Company employed during the project are categorized below. The results of the challenges and issues associated with IT Company followed procedures to use the Scrum methods in development projects are examined.

5.1 Scrum methods

5.1.1 Product Backlog

The Product Backlog contains everything the team might ever work on. The product owner prioritizes the work on the backlog for the development team. Afterwards, the development team uses the prioritized backlog as its single source of truth for the work that needs to be done.

The following quote is from the product owner,

"There was a lot of conversation between us and Tour Operator’s key persons before it ended up on the product backlog. We did not get all the details at the beginning of the project because we knew that we would get more information during the development progress."

As expressed above, if you don’t apply plan driven development to your project, you don’t need to get all the details before the project starts. In the Scrum process, the use of documentation is limited compared to traditional methods. The process does not give much importance to the specification of documents from the clients.

The following quote is from one of the developers, “Defining backlogs is a problem; they are not always well explained. This was a problem for us because many scenarios and details were unclear during the project and we had to ask the product owner each time things were unclear”

Version One tools were used by IT company during the project development. Through Version One you can plan, prioritize and communicate your strategic, cross-project initiatives with downstream traceability to ensure the strategy aligns with the execution. All the main tasks were collected and prioritized.
5.1.2 Backlog Refinement Meeting

The goal of Product Backlog refinement is to work with the Scrum Team and stakeholders (when relevant) to get Product Backlog items clear enough, so they understand what the stakeholders are asking for and why they are asking for it. Backlog refinement may need the information so it can break the higher priority large items into smaller items. This implies a need to know whether things are large or small.

Working from a prioritized backlog of small user stories allows a team to get value and high-quality feedback at frequent intervals. Many teams struggle to split large user stories into small stories in a useful way. IT company had a product backlog meeting before the team started sprint planning. During the product backlog refinement meeting the team and product owner discussed the top items on the product backlog. The team was given a chance to ask questions that would normally arise during sprint planning:

- Why do we need to create a flight optimization?
- Are all users allowed to access this part of the system?
- What happens if the flight arrival time is 00:30. Will the arrival date be counted as +1 day?

The product owner may need some time to answer the questions prior to presenting the story to the team. During story grooming, the question is will a story of that a size be fitted into a sprint, or, should it be broken down into smaller increments.

The following quote is from one of the product owners, "Our Epics are estimated in t-shirt sizes. User stories are point based and task
level is based on hours”. This is where the story may be reviewed for the first time so we make sure that the team understand. We acquire additional details/relevant discussion, acceptance criteria and any technical risks and any unknowns. We have a light technical discussion and after that we t-shirt size this. Basically a S, M, L story will most likely fit into the sprint (story should fit into a sprint) and if it’s an XL I have to consider splitting it up or I know that this is more than one sprint worth of work”

There are two levels of estimation in Scrum method: Product backlog estimation and Sprint backlog estimation. Product backlog estimates and provides visibility of each user story; they are high-level estimates but less accurate compared to sprint backlogs. Most challenging in the Scrum method is the estimation of backlogs. The developing team lacked an estimation of the stories. They did not even know what kind of challenges would they have during to the development of tasks.

The following quote is from one of the developers, "We did not have much knowledge about agile and scrum. This made us a little bit nervous when having meetings. We have only learned about scrum from videos on youtube”

Some of the large stories were divided into different stories in the refinement meeting. Below figure 5.2 shows how creating flight stories were divided into different tasks.

Splitting Stories

![Diagram of splitting stories]

Figure 5.3: Splitting Stories
5.1.3 Sprint Planning Meeting

The purpose of sprint planning is to select a set of product backlog items to work on and have a rough idea of how to achieve them. Achieving this does not require the team to know every task they'll perform. Sprint planning might want a more accurate idea of how large or small the highest priority items are, so it can give a list of those expected to be finished in the upcoming sprint. During the sprint planning meeting, the product owner describes the highest priority features to the team. The team ask enough questions so that they can turn a high-level user story of the product backlog into the more detailed tasks of the sprint backlog.

The following quote is from the scrum master:
"We reviewed the groomed story and if there were no changes we were ready to Story point. If we had additional information we discussed this and then the story point. So essentially at the time of commitment you get as any additional new details as possible and you commit to the story. Story point in turn will determine the velocity and commitment for the sprint”

The team chose stories from the product backlog to develop these in Sprints.

The following quote is from one of the developers:
"In the first sprint we selected some stories from product backlogs that we believed we could finish in the first sprint delivery time but we understood that we had chosen too many stories and we could not deliver them in the first sprint” This shows a lack of knowledge regarding estimation of the sprint may cause late delivery of the project.

5.1.4 Sprint Backlog

The goal of the Sprint backlog is to create an increment of functional software at the end of this sprint. The software company needed to deliver value and product to the customer at the end of each sprint. The product owner requested that they work on the highest priorities first as they deliver the most value up front. The team chose stories from the product backlog and moved these to the sprint backlog. When they moved these stories they had to consider that the estimation of each sprint would not exceed four weeks.

They started with prioritized tasks that they moved from the product backlog to the sprint backlog. The first sprint estimation time was three weeks. During the project development there were some parts of tasks that were undefined and many scenarios were unclear and they had to ask product owner. IT team was working against time. They had to deliver a potentially shippable product increment.

The following quote is from one of the developers:
"We had to ask the product owner to explain some scenarios because some scenarios had not been considered. For instance, one of the tasks was to create
flight with both to and return. But when you create flight in the system you also need to create optimization of flight seats so that the flight cannot be overbooked”

Delivery of shippable product increments took more time than estimated. The first sprint took five weeks whereas the estimation of the first sprint was only three weeks. This occurred because the developing team did not clearly understand the tasks and had to ask the product owner about each task in turn. The product owner was in continuous communication with Tour operator staff that were involved in the project. Hopefully they were sitting in the same room making it easier to ask questions than using e-mails or telephone calls!

After delivery of the first shippable product increment, Tour Operator found a number of bugs and missing parts and they informed IT company about the bugs. Certain parts were missing because they had not been developed as Tour Operator had not provided the necessary details.

Figure 5.4: Velocity Trend - Version One
According to the Scrum guide the client can adjust the requirements throughout the development process. Normally, the sprint will take three to four weeks but if the client changes the requirements during the sprint it will be difficult for the product owner to re-do the estimation to complete the task within the given time. This proved to be the case for IT company who needed more time to fix the bugs and perform the necessary adjustments according to the changing requirements of Tour Operator.

Figure 5.5: Sprint Backlog - Version One

5.1.5 Daily Scrum meeting

Communication is an important factor for the success of a project. One of the best ways to spread information within an organization is through meetings. Daily meetings address the commitments of the team: e.g. on that day they are committed to do X, Y and Z. They are about tracking A, B and C from yesterday. The whole team meets every day for a quick status update. They stand up to limit the meeting to about 15 minutes. In this manner one can identify:

- What was done yesterday?
- What will be done today?
- Are there any impediments in the way?

According to interviews the scrum team had some challenges with the daily scrum meeting. I had the opportunity to observe some of daily meetings and saw that some of developers came late and the meetings took more than 15 minutes. According to the principles of Scrum, the scrum team should meet on time, always in the same place, and the meetings should always start at the same time. Can we perhaps expect tolerance for delays? If it happens once it should be a tolerated but if it happens frequently the Scrum Master (SM) and Dev Team should correct it!

Maybe the SM should start by finding out *why* the person arrives late. Is it unavoidable e.g traffic problems and the cost of starting out earlier...
would be prohibitive? Is it due to poor organizational skills i.e he/she wants to be punctual but struggles to keep time? Is it due to wrong attitude i.e he/she doesn’t care that the team is affected by his/her behaviour.? Maybe he sees no value in the daily scrum and is pushing for something to change. Or are there other reasons?

The following quote is from the SM

“Our meeting took more than 15 minutes. Developers started to talk in more detail about what they did yesterday and what they had planned for today, and it was difficult for me to coordinate the discussion. This is due to the developers’ lack of knowledge concerning the scrum guide and agile.” As the SM mentioned in the above quote, the daily meeting took more than 15 minutes. This shows that the SM lacked leadership skills. The SM needed more training regarding how to manage the daily scrum. The rules of Scrum are simple and complete, but it’s easy to stray from the plan that Scrum has so clearly laid out. Part of the role of the SM is to teach and coach during the Scrum and make sure that the framework is being followed. It’s simple but not easy. There is always the temptation to go back to old habits and make exceptions to the rules.

5.1.6 Sprint Review Meeting

As described in chapter 2 The Product owner, Development Team, Scrum Master and Stakeholders(optional) take part in the sprint review meeting. The team demonstrates their product increment. This meeting can take up to 2 hours for a two week sprint. The team demonstrates the potentially shippable product increment and Product owner declares what is done. In this meeting Stakeholders get an opportunity to submit feedback. People usually don’t know what they want until they see it working. New tasks/changes that come from feedback, must be added to the Product Backlog.

In the case study, Tour Operator’s key persons were invited to the Sprint review meeting by the Scrum Master. In this meeting IT company were going to present what they did and get feedback. According interviews this meeting took more than 5 hours. As described before first sprint was planned three weeks and for three weeks sprint, sprint review meeting should not be over four hours.

The following quote is from Project Manager from Tour Operator company

“In this meeting IT company presented what they had done. When we checked we realized that the Flight module was not exactly that we requested. We needed to give them more feedback for it”
We see that there was a lack of communication, and misunderstanding, between Tour Operator (TO) and IT company.

In this scenario we see one of the advantages of agile; at the end of each sprint the project priorities are evaluated allowing clients to give their feedback so that they ultimately get the product they desire.

5.1.7 Sprint Retrospective Meeting

The purpose of the sprint retrospective is to evaluate the previous sprint and to improve the process. Improving the process is really important and also promote better communication between team members.

The following quote is from the SM
"In the retrospective meeting we discussed

- What have you done well?
- What could you have done better?
- What have the team done well?
- What could the team have done better?

and after point 4, we discussed how the team could have performed better and how to improved, defining actions based on point 4 only. This procedure was done step by step, with 5 minutes between each step. The reason for this was to:

- Make the person feel good about himself/herself if he/she worked well, or slightly unsatisfied otherwise.
- Identify how he/she can improve individually.
- Point out what worked well internally within the team and give credit to the others for what was positive.
- Identify how we could improve the team, considering that the team is a set of individuals that have to perform and improve individually and together.
"

In the retrospective meeting the atmosphere is very important. If people don’t feel it’s safe to talk, the retrospective meeting will not be beneficial.

The meeting might be boring and monotonous for some teams, so changing the location from time to time may be a good idea.

As previously mentioned in section 5.1.4 problems arose after each incremental delivery. Bugs were found that testing prior to delivery had not exposed and TO realized that IT company’s testing before user acceptance testing was not optimal. There were also requirements that were not fulfilled in each delivery and these were postponed to the
next iteration. Another big problem was the performance. After each incremental delivery the Tour operator system started to respond slowly. Initially the system was responding very well but when it became more complex its performance declined.

As a result of this IT company committed to completing the project in ten months. However the project was delayed five months; there were many reasons for this delay. The Scrum team had challenges when practicing agile such as slowness, resistance to change and over eagerness to make changes. Other big challenges were wrong estimation, wrong prioritizing of the tasks and the absence of a risk plan. They did not consider what kind of risk they would have during software development. All these factors show that the team/organization lacked knowledge about Agile. I will provide more details about the challenges and issues during software development in the discussion chapter.

TO lost its confidence in IT company because repetitively made incorrect estimations and were slow to change the requirements. The cost for TO company was thousands of euro. TO company were planning to use the new software for their summer season and had created all their summer products on the new system. In case of any unexpected situation they had fortunately created all hotel contracts on the current system in addition and they used the current software for the summer season as well.
Chapter 6

Discussion

The purpose of this study was to find out why agile development is advantageous in the travel industry and elucidate the challenges in relation to Agile’s methods as applied by IT company.

This research was conducted with a single case study approach and data was collected through face to face interviews with participants and key informants from Tour Operator company and IT company. The case study helped me to understand how poor quality can bring down a software development project.

6.1 Implications for Research

In this section the implications of this research within respect to body of knowledge already established in the area will be discussed. Agile project management is emerging in the field of project management, developing a new way of thinking. There is ongoing research being conducted in the area, and hopefully this study will contribute new knowledge to the field of project management in the travel industry.

In 1995, Steven L. Goldman, Roger N. Nagel, and Kenneth Preiss, the authors of Agile Competitors and Virtual Organizations (Van Nostrand Reinhold, New York), offered this definition of agility: “Agility is dynamic, context-specific, aggressively change embracing, and growth-oriented. It is not about improving efficiency, cutting costs, or batten down the business hatches to ride out fearsome competitive “storms.” It is about succeeding and about winning: about succeeding in emerging competitive arenas, and about winning profits, market share, and customers in the very center of the competitive storms many companies now fear”.

In Agile project management you don’t base your daily work on a long-term plan; you organize your project into segments and each segment is worked on with relative autonomy. In agile development, all stakeholders
and the end user are actively involved in the process.

People can transfer ideas faster by talking face to face than by writing and reading documents. A few designers sitting together can produce a better design than each could produce alone (Highsmith J. and Cockburn A., 2001). So, all projects begin with a conversation and then maybe there’s a little brainstorming. As to how to proceed from there, personal and cultural biases probably have more to say about what direction the group goes in than any rational set of decision criteria.

To make a rational choice one must be familiar with more than one path and the company must allow different paths to be taken. However, in many companies the path is already decided.

One factor that characterizes the first decision to be made is the project’s level of predictability. If the path from start to finish is clear (a set of unambiguous instructions can be made that will take us from start to finish without error) then the direction of traditional project management (the right side of the Agile Manifesto) will be appropriate, but if it is largely unpredictable, the Agile direction (the left side of the Agile Manifesto) is appropriate.

According to Agile Manifesto "working software over comprehensive documentation". That is correct, however there should be a short list of requirements so the team can understand the scope of the project and understand exactly what the customers want to include in their system. It is not easy to create full documentation of the requirements because the market changes quickly and people’s requests often change. It is also time consuming to prepare full documentation of requirements. This is particularly common in the travel business where requirements frequently change. One must follow the market and competitors or the ongoing project may not bring added value to the business. This demonstrates why traditional software development is unsuitable in the travel industry.

Any company that values abstracts of customer and employee satisfaction, short work cycles, staff work, efficiency, and speed could crucially benefit from incorporating Agility into their company.

As described in previous section, Agile development aims for rapid development, customer collaboration, business value and quick responses to change and continuous development. One of the important elements of agile project management is that it is able to cope with requirement changes, even when they occur quite late in the process. The customer has frequent and early opportunities to see the work being delivered and to make decisions and adjustments throughout the development project. The customer gains a strong sense of ownership by working extensively and directly with the project team throughout the project. If time to market for a specific application is a greater concern than releasing a full feature set at the initial launch, Agile can quickly produce a basic version of working software which can be built upon in successive iterations. Development
is often more user-focused, probably due to frequent communication and interaction with the customer.

Several companies have increased their productivity simply by choosing to adopt Agile methods. A few of the most important factors that characterize Agile management are; managing change, more utility for the client during the project, a motivated project team, an even and reasonable workload, an airplane cockpit for projects and that it works in many businesses. (Gustavsson, 2011)

When Gustavsson (2011) says more utility for the client during the project he means that because the project is divided into cycles of one to four weeks and that at the end of each of these cycles something will be delivered to the client he or she is forced to be involved and make decisions. Through these constant checkpoints between every cycle the clients do not have to worry about whether the project is going in the right direction or not since he or she is constantly updated.

Agile requires good skills such as domain, technology and fairly good knowledge of agile practices. Additionally of course, knowledge of the right techniques for the project and the ability to choose among them is important. Without direction, a team asked to discard waterfall methodology would simply devolve into "code and fix". That’s chaos and not agile.

### 6.2 Implications for Practice

This section will discuss some of the implications of the research with respect to project management policy and practice.

As described in the previous chapter, before embarking on a project these aspects should be considered:

- Project size
- Customer Availability
- Level of integration with external systems:
- Customer tolerance for scope and cost changes
- Time to market

IT company follows all the ceremonies, artifacts and the lifecycle of Scrum but they are not following the exact way of using the method.

There are some common challenges in practicing agile scrum methodology in a team;

Firstly, to get off to a good start an experienced Scrum Master or Agile Coach is needed. Many companies perform organizational changes and
think they can proceed having read a few blogs or attended a short course. This is, however, insufficient. The SM must have the necessary leadership and coaching skills to help the team when they encounter problems. It is therefore important that the SM himself / herself receives sufficient coaching in order to perform his / her job satisfactorily.

Secondly, those involved must be educated about what agile really is and is not. For example there is no such thing as "agile scrum methodology"; there is agile and there is the Scrum Framework, but both serve different purposes within a common theme.

Finally, the hardest part about implementing agile is helping people to *really* understand it and, in turn, to change. Although it may sound simple, people generally struggle to understand the concept at first. When they finally grasp it, change occurs. Experienced coaches know how to persist using different techniques to help teams and management understand and change.

In my case study the interviews and survey (see Appendix B) show that IT company encountered challenges when it adopted Agile. The development teams had limited knowledge about Agile/Scrum which they had acquired by watching a few videos about Agile software development and Scrum methodology. A newly built agile team needs training to understand how scrum is different from traditional methods. Team members' roles within the team vary. Agile teams should be "self organized", a concept that is not easy to grasp. Also, it’s difficult for a new agile team to understand that they are now part of the decision making process and have more responsibility for the decisions made. Acquiring sufficient finances to provide the necessary training may be problematic, but if the team doesn’t understand agile development, it cannot be expected to somehow be agile by magic! Without that understanding, the project will fail; it’s almost impossible to invent agile as you go along. In addition, training the team in Scrum without training them in Agile is also doomed to failure - it will just take a little longer to fail. Scrum is merely a framework and training within the context of the framework and also in the specific practices used to realize Scrum is needed. Typical Scrum training is not sufficient either. Finally, you need to train whoever made the decision to proceed with Agile without the necessary up-front education. Unfortunately the SM struggled to fulfill his role and was unable to coach the team members sufficiently and aid their understanding of their professional discipline and the basic concepts such as accountability, assertiveness, teamwork, shared leadership, etc. and why these were important for their team. The team should understand that there is no such thing as "agile scrum methodology"; there is agile and then there is the Scrum Framework and both serve different purposes within a common theme.

If you have a team that is new to agile, the project risk can be reduced by making the first couple of sprints easier. For example,
if 50 features are planned for the first six sprints and a product will be ready for implementation at the end of the third sprint, it would be wise to make the first sprint lower risk by working on easier, less complex features. Assuming the first sprint was successful, more difficult features can be addressed in the second sprint. If the team requires more practice at working in an agile environment then the features should be kept more simple. Any features that require a lot of cross department communications, advanced technical capabilities, or are extremely intricate to produce, should be moved to a subsequent sprint. Allocating features to sprints in this way reduces the overall risk to the project by providing an opportunity for the team to adjust to agile techniques, work on being a collaborative team, and to build their confidence.

Communication is an important factor for success. From the very start of the project it is vital that there is good communication, between all those involved including team members, managers, project sponsors, clients, valued users and stakeholders. A lack of communication can lead to misunderstanding of the requirements. Therefore, good communication and collaboration is the key to success.

Looking back at the case study we see that PO added some requirements during the ongoing sprint which inevitably presented a challenge. According to Scrum "rules", once a sprint begins, the new requirements or stories cannot be added to sprint’s backlog. The reason for this is that the time box i.e. the amount of time set aside for the sprint is, of course, related to the amount of work to be completed in the sprint, in the form of specific requirements that have been estimated (“Scrum Rule,” 2016).

"A late change in requirements is a competitive advantage.”

Mary Poppendieck
6.3 Answering the Research Questions

The objectives of the thesis have been obtained. The research questions are answered as follows:

6.3.1 Why do we need agile development in the travel industry?

This was answered through the literature search in order to identify the advantage of Agile software development.

A few decades ago, the waterfall response was, in fact, adequate because new markets were not being created ever day, the Internet did not exist as we know it today, and it took much longer for people to spread their messages and ideas around the world. (Viscardi, 2013)

Agile methodology saves time and money. To prepare detailed documentation is time consuming. The Agile Manifesto prescribes that the agile team should value working software over comprehensive documentation. However, that does not mean one should not create documentation; one should create documentation that provides value. Writing detailed documentation at the beginning of the project has many drawbacks. The documentation will be out of date by the time it is completed. In the travel industry requirements change rapidly so the team needs to be able to adapt to changing requirements. Software development is actually a process of design from beginning to end. It includes design work to accommodate requirements whose details become clearer as the project progresses, and design work to reflect what’s learned about the tools and components used to develop the software. The process of software development is one of continuous design, and therefore of continuous change (Stepanek, 2005).

Agile methods have been very successful for some types of system development such as:

- Product development where a software company is developing a small or medium-size product for sale

- Custom system development within an organization, where there is a clear commitment from the customer to become involved in the development process and where there are not a lot of external rules and regulations that effect the software (Sjøberg & Lindsjørn, 2014)

My case study shows that there was a clear commitment from Tour Operator and they were involved each development process so there were not many external stakeholders. Tour Operator changed some requirements during the development process and wanted to see the product incrementally. As described in chapter 2 Agile software development methodology
focuses on customer satisfaction, an iterative and incremental development approach, a bug free product and acceptance of changing requirements.

Looking at my case study example, it can be understood that customer is extrovert and there are some ambiguities regarding the requirements in the project. Using Agile project management during uncertain situations regarding the requirements is advised.

Let’s look at another successful Agile software development in the travel industry. Sabre’s recent revival of its 25-year-old air-travel reservation system shows how agile practices can improve the outcome of software projects: “Sabre has tried to overhaul its reservation system before—most infamously from 1988 to 1992, when it spent 125 million dollars on a mega project to do just that. A few weeks ahead of the promised completion date, Sabre had to junk the entire system (Stepanek, 2005).

So what makes this new four-year, 100 million-plus dollars success different from that old four-year, 100 million-plus dollars catastrophe? This time, it wasn’t a big leap. Borrowing techniques from so-called agile programming, Sabre did it as a series of small steps. . . . As a result, the system already looks a lot different now than in 2001, when the project started. Small steps and a willingness to change direction make that possible (Hayes, Frank., 2004).

I have worked in the travel industry for 13 years and managed two big projects in this field. During project development we sometimes had to changed our requirements because we needed to follow our competitors and the market. We wanted to see the product incrementally to check everything was proceeding correctly and as requested. Preparing a full requirement document is time consuming for us and we were unable to provide all the necessary details before the project.

6.3.2 What are the challenges in relation to the Agile methods?

To answer this research question I will first describe common challenges and then compare with the challenges and issues associated with the IT company following procedures to use the Scrum method in development projects.

There are some common challenges related to the Agile method

Working remotely and communication

One common challenge is the pressure on team members work remotely. The most effective form of communication (for transmitting ideas) is
interactive and face-to-face, as at a whiteboard. People sitting near each other, with frequent, easy contact, will develop software more easily; that is, the software will be less expensive to develop. (Cockburn A., 2000) As Cockburn A. (2000) says face-to-face communication increase productivity. When we look at my case study we see that the Agile team worked in the same location, even in the same room. This was beneficial to communication as they worked face to face; they did not even need to investigate communication tools. So for IT company, working remotely and communication was not challenge.

**Self-organizing teams**

Agile teams don't have managers they are self-managed. If there is a "manager," then they should have a supporting role for the team. Traditional management that involving time management, estimates, deadlines, telling people what to do, analyzing people’s performance, etc. has no place in agile. Self-organizing teams is a challenging concept, especially when moving from traditional project management to Scrum. Unlike traditional teams, the self-organizing empowered teams are not directed and controlled from the top; rather they evolve from team members participating actively and collectively in all the Scrum practices and events (“Self-organizing Scrum teams - Challenges and Strategies,” 2016).

It is difficult for a new agile team to understand that they are part of decision making group and have to get more responsibility for decision made by them or decision make by their agreement. This was not a problem for IT company; the agile team was self-organizing, managed itself and had autonomy. It was not directed and controlled by the top management.

**People Mindset**

Frequently, people working on Agile projects, do not understand their responsibility and the thought process for a successful project. People resist adopting new values and principles as it demands a change in mindset. Changing the mindset of agile teams is always a bit difficult. It is easier to change people than their minds. If you want people to follow ideas, make it easier for them to learn and use, or they will never change. It’s very hard to persuade people to be Agile if they do not just feel it from the start; they must be shown how. This was a challenge for IT company. Their team was always criticising the changing requirements and were not willing to make last minute changes to requirements after delivering a sprint. The interviews revealed that when the requirement were changed developers’ motivation dwindled.


**Inability of the team**

If you decide to follow Agile principles then you should be careful how you choose your project team. Some agile teams don’t know the difference between practicing Agile and being Agile. Inability of the team may cause the project to fail. The team will be unable to understand and appreciate the agile values and principles such as early feedback, customer collaboration etc and if they are unable to understand, they will be unsure why they should practice them. Eventually the team will become non-agile.

**Experienced Scrum Master**

When an organization decides to convert to agile but the teams have no or little experience in agile methods, the SM plays a crucial role. He / she is not just the facilitator but also the coach for the team. If the development team has little knowledge about scrum it is SM’s job to improve this. The challenge is for the SM and scrum coach to have enough expertise and communication and, coaching skills to provide this knowledge to the team.

**Experienced Product Owner**

The PO is typically a project’s key stakeholder. The PO is responsible for prioritizing the backlogs and choosing the requirements for the next iteration. An experienced PO needs to understand requirements properly and correctly prioritize the product backlog. In my case study PO did not prioritize the product backlog correctly and added more requirements during the sprint. This was a problem for IT company and was one of the reasons for late delivery of the software. According to the Scrum guide, adding requirements during the sprint is not allowed. If the PO has new requirements then they need to be added to backlog and discussed in the next sprint meeting.

**Organization’s culture**

A work with a high level of discipline which respects organizational hierarchy may have difficulty applying Agile methods. Ideally, when wanting to adopt agile methods, then working place should be a free working field with open communication irrespective of people’s seniority. In Agile, communication is an important factor. Organizational culture is considered to be a factor effecting successful adoption of an agile method (Diane E. Strode & Tretiakov, 2009)

IT company had discipline but not at such a high level and did not encounter this problem.
Challenges with user story creation

Another common challenge is when PO doesn’t know what he / she really wants and doesn’t realize what is really needed. Challenges might occur because of keeping the stories non-technical and keeping the stories small enough. In the case study IT company experienced challenges with the user stories. Some user stories were unclear to understand and were short. Remembering that a user story is not a contract but instead a placeholder for a conversation. Many teams think user stories are another way of writing requirement ‘contracts’. Conversations are more important. In my case study the Scrum team solved this challenge by face to face conversation. Another problem IT company encountered was that the of adaptability even though they followed Scrum practices.

Estimation

Estimation is one of the challenges of agile projects because requirements frequently change which requires the PO to recalculate estimation. According to Martin Fowler (2013) estimation is valuable when it helps you make a significant decision while Duarte (2015) claim that estimates do not provide more value to the customer than inventories. Whether estimation is valuable or harmful I think we consider the purpose of the estimate. The team should definitely estimate their work before they start. In agile methodology, the team is the core component. It is the team members who will size the work by estimation and based on the estimation can say how much work they can commit to. Estimation by the team will really help the team members in addition to product management for the prediction of releases. When doing Scrum the team’s velocity must be known, so some kind of estimation is needed in any sort of agile development, or even in non-agile development.

At the same time, estimation is valuable when we need to understand the likely costs in advance, so that we can decide whether it is worth the investment. The customer may also ask when the project will be delivered as in my case study example. Miscalculation of the estimation make may result in an unsatisfied customer. In my case study estimation was a challenge; IT company miscalculated estimation and the customer was unsatisfied. Since requirements change frequently in the travel industry, estimation should only be performed regarding the cost of the project and the team members’ velocity. Otherwise miscalculation may lead to different results.
A famous quote from General Dwight D. Eisenhower: "In preparing for projects I have always found that estimates are useless, but estimating is indispensable"; we still need to do it, whatever we call it and whatever we’re counting when we do it. (Morris, 2013)

6.4 Recommendations

There are some recommendations I will make based on what has been discovered during the case study. As previously described, communication and collaboration are the most important factors in agile project. As IT company if you are going to work with a customer who knows the business well, prepare a test-driven design together and prioritize requirements together. This will help the developers to understand the project correctly.

Another recommendation concerns training of the agile team. The team should understand agile methods properly. Training is critical, but is itself not enough. The team must learn to think in an agile manner, learn to work collaboratively and learn how to manage itself.

6.5 Limitations

Single case study analysis has, been subject to a number of criticisms, the most common of which concern the inter-related issues of methodological rigour, researcher subjectivity, and external validity. With regard to the first point, the prototypical view here is that of Maoz (2002), who suggests that “the use of the case study absolves the author from any kind of methodological considerations. With single case study research the researchers often use information-oriented sampling thus the information is not very wide and does not provide deep insight.

The case study was not completely agile and IT company did not claim that they were completely agile either. They tried to follow agile principles but they lacked experience with agile methods. Therefore, I can not claim that these finding are transferable to all agile projects.

I have only been able to study one case, that of Tour Operator software management. Studying only one case may lead to less reliable results than if more cases had been studied. Other travel industries such as air plane software management could have been investigated for greater validity but case studies are is time consuming to conduct.
6.6 Future work

Future work on this subject could be multiple cases study of different sized projects and in different travel companies to see which project management methodology they used in their software development and how they chosen which methodology was suitable for the project.

It would be also interesting to investigate a larger project that has applied Agile methodology in order to see whether they encountered the same problems seen in this case study and how they chose to solve them.

During the study I have focused mainly on Agile Project management and its framework in the travel industry. This is because I am managing a project in this area. It was easy for me to find a relevant project with the relevant people and therefore easier to study. During the study I have seen how they manage the project and apply Agile Project Management.

I have gained some insight from the interviews and attended some meetings for observation but It would have also been interesting to go into each of the different Agile methods in greater detail.

I would have liked to research a complex airline software development and Airport Baggage Handling System. This is something that requires further investigation and would have been interesting to look into.
Chapter 7

Conclusion

Two research questions were sought answered in this study. The goal of this thesis was to find why Agile project management should be used in the Travel Industry and what kind of issues IT company had when they applied scrum to the Tour Operator’s software project.

In this research agile software development methodologies have been examined from the following perspectives: the motivation behind their appearance when compared to traditional (Waterfall) project management. Agile is a great technique to use when business needs are frequently changing or when the business wants to receive product benefits earlier. With Agile you can focus on what the business needs and if this changes the new business needs can be accommodated in the next iteration. Agile is frequently used to manage IT projects, but can also be used to manage non-IT projects.

All Agile practices share the common concept of producing working software in multiple iterations with teamwork, quick response to change, better customer collaboration and minimal documentation. Managing software projects in small iterations and with continuous feedback minimizes the project management overheads and risk of failure. Successful agile software development requires that business people and developers work together daily throughout the project.

In the Travel industry requirements changes frequently during the time the team is developing the software because market conditions are continuously changing. Creating a full requirement document is time consuming and not necessary since requirements often change.

The data analysis showed that IT company’s scrum team lacked knowledge about agile and scrum resulting in incorrect estimation of the project and late delivery. The SM had theoretical knowledge about Scrum but not practical experience and lacked leadership skills.

The philosophy behind Agile is simple but practicing it may be difficult. The best way is to have an Agile coach assist the team. Getting people trained at various levels by having an agile champion in the organization, evangelizing agile, having workshops and coaching SM is
a key factor to Scrum adoption. (Hazrati, 2008)

A SM will often need to coach and prompt the team members more during the first few sprints when using Scrum. As time goes by, the team will understand how they can self-organize to accomplish the work. In most organizations management roles still exist. These people are responsible for establishing the boundaries within which teams operate, removing organizational impediments and often have "people care" responsibilities.

The SM is supposed to be a servant leader, meaning his / her role is responsible for supporting and protecting the team. The hard work of delivering the product is primarily the responsibility of the team as guided by the PO. I’ve found that a SM who knows little about the context or the technology is more effective than someone with deep understanding of either or both as it keeps the SM from meddling with the team.

Agile is a very good ‘method’ to follow as long as you have a team of experienced programmers and a good technical lead. If companies want to apply agile development principles and practices they need to understand the value of Agile. Without a solid understanding of the agile development process and its related techniques, successful implementation of agile methodologies can be difficult.

It is difficult to find the best framework but elements can be added/subtracted from a methodology to find a framework that fits the specific context. In my experience, if develops software for a tour operator I would suggest the use of agile methods. First start with Scrum and create your version of XP such as pair programming, test driven development, automated testing and re-factoring.

My final word to you, reader of this thesis anybody who has a complex project can benefit from using Agile. Any project needs resources, and people with experience, authority and commitment to see it through. Agile does not guarantee any of these things, but what it does is help you to focus on producing things of value to the customers and business.
References


Peter Trudelle. (2013). What is Agile, a framework or methodology?
Tracy Ireland and John Schofield. (2015). Information Technology and
Tourism.
//en.wikipedia.org/wiki/Scrum_(software_development)
Appendices
Appendix A

Interview guide

This study aims to increase the understanding of the challenges related to the projects’ development process and technology while using the Scrum method. The objective is to define the challenges you are facing and possibly identify ways to deal with them in the future. We will talk to you about your work routine and the problems you may be facing. This discussion will be organized in 4 themes: your personal background, your daily work, how do you use the Scrum method, and the challenges you face during the process or with the technology used. If you feel uncomfortable with any question we can rephrase it or skip it. All the interviews will be conducted anonymously and all the information collected will be used for research purposes only. The interviews will be recorded, if you don’t feel comfortable with this please let us know.

Interview Questions:

Theme 1: Personal background.

1. What is your role in the company?
2. How long have you been working with this company?
3. What kind of background do you have?
4. What are your previous work experiences?
5. How long have you been working with the agile?
6. Which method have you been working with before?

Theme 2: Daily work.

1. Can you describe a typical working day? (Working time, day activity)
2. Do you usually work alone or in pair/team? (Programming)
3. How big are the teams you are working in/with?
4. Is your work in contact with the clients or within the company exclusively?
5. The projects you are working with are for the customers or are internal projects?
6. How many meetings you have to attend, how long do they take, how productive you think they are?
7. What do you find more challenging in your work day? And what can be improved?
8. Can you describe something that you find exciting about your role in the project?

**Theme 3: Specific using of agile methodology.**
1. Do you think agile is suitable for this project? Why?
2. Is it difficult for you to find clients for the agile projects?
3. What does the agile methodology adds to your work/projects?
4. Who are involved in a Agile team? Can you describe the roles?
5. Which members of the agile team are you most in contact with? [Why? What do you do together?]
6. What kind of projects you are handling? Are they short or long term projects? are?
7. Are you usually able to deliver the project on time? If not, how long is the average delay?

**Theme 4: Technology and process.**
1. Can you describe the current project development process?
2. How satisfied are you with this process?
3. In your opinion how your clients are satisfied with this process? Why?
4. Which of those phases you find more challenging and why?
   Defining product backlog/sprint backlog;
   Execution of sprints;
   Testing and delivery or rework;
   Organizing effective meetings;
   The technology used.

5. What challenges do you face, in relation to the process (procedure)
of agile methods in your projects?

6. What challenges do you face, in relation to the technology (tools and
   techniques) of agile method in your projects?

7. Have you ever faced issues with the client in relation to the process
   and technology? If yes, what are they and how you dealt with them?

8. How the client’s involvement in the process does affect your work?

9. If you can change one thing in this process what would it be? Why?
Appendices
Appendix B

Survey guide

Survey Questions and result

This survey was performed in April 2016 and there were 9 respondents. The survey was sent out to the Scrum team involved in the Tour Operator software project. The goal of this survey was to attain information about what kind of knowledge team had about agile and agile methodologies and whether they were satisfied with agile in the project.

I’m sharing the results, and in particular the source data, of my surveys for several reasons:

• Other people can do a much better job of analysis than I can. If they publish online.
• I think that it’s a good thing to do and I invite others to do the same.

Here are the survey questions

1. How many years of experience in the Software Development area do you have?
2. What is your job role?
3. What is the highest level of education you have completed?
4. How would you rate your knowledge of agile methodologies?
5. Which agile methodologies have your organization adopted?
6. Which agile techniques have your organization adopted? (check all that apply)
7. How have agile approaches affected your productivity?
8. How have agile approaches affected the satisfaction of your business stakeholders in the work produced?

9. How have agile approaches affected the quality of the systems produced?

Here are the survey results.

![Survey Q1 Result](image)

**Figure B.1: Survey Q1 Result**
Figure B.2: Survey Q2 Result
Figure B.3: Survey Q3 Result
Figure B.4: Survey Q4 Result
Figure B.5: Survey Q5 Result
Figure B.6: Survey Q6 Result
**Figure B.7: Survey Q7 Result**

*How have agile approaches affected your productivity?*

<table>
<thead>
<tr>
<th></th>
<th>Much lower</th>
<th>Somewhat lower</th>
<th>No change</th>
<th>Somewhat higher</th>
<th>Much higher</th>
<th>Don’t know</th>
<th>Total</th>
<th>Weighted Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no label)</td>
<td>0.00%</td>
<td>0.00%</td>
<td>12.50%</td>
<td>50.00%</td>
<td>37.50%</td>
<td>0.00%</td>
<td>8</td>
<td>4.25</td>
</tr>
</tbody>
</table>
How have agile approaches affected the satisfaction of your business stakeholders in the work produced?

Answered: 8  Skipped: 1

<table>
<thead>
<tr>
<th>Response</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Much lower</td>
<td>0.00%</td>
</tr>
<tr>
<td>Somewhat lower</td>
<td>0.00%</td>
</tr>
<tr>
<td>No change</td>
<td>0.00%</td>
</tr>
<tr>
<td>Somewhat higher</td>
<td>26.00%</td>
</tr>
<tr>
<td>Much higher</td>
<td>76.00%</td>
</tr>
<tr>
<td>Don't know</td>
<td>0.00%</td>
</tr>
<tr>
<td>Total</td>
<td>8</td>
</tr>
<tr>
<td>Weighted Average</td>
<td>4.75</td>
</tr>
</tbody>
</table>

Figure B.8: Survey Q8 Result
How have agile approaches affected the quality of the systems produced?

Answered: 8  Skipped: 1

| (no label) | 0.00% | 0.00% | 0.00% | 37.50% | 62.50% | 0.00% | 8 | 4.63 |

Figure B.9: Survey Q9 Result