Application and evaluation of methods for merging user experience design with agile software development

Mikael Eriksson Vikner

January 19, 2016
Master’s Thesis in Interaction and Design Technology, 30 credits
Supervisor at TFE-UmU: Håkan Gulliksson
Examiner: Thomas Mejtoft

UMEÅ UNIVERSITY
DEPARTMENT OF APPLIED PHYSICS AND ELECTRONICS
SE-901 87 UMEÅ
SWEDEN
Abstract

Cinnober is an organization that develops advanced software solutions for financial institutions. As a part of the technology toolkit used at Cinnober there is a web framework with which GUI development can be driven from the data available on the server, through configuration rather than development. Rather than having the user interface emerge as a result of technology and available data, they would like to explore a software development model driven by user centered design. Cinnober practices scrum, an agile software development framework, which has proven difficult to integrate with user centered design.

This thesis strives to identify suitable methods for performing user centered design in the environment of agile software development. A development process based on scrum, lean UX, staggered sprints and the effect map was then utilized and evaluated in a short development project at Cinnober.

Utilizing and evaluating those methods yielded valuable input which can be of use in future development efforts. While there was plenty of positive feedback from the development team there was also some room for improvement. Additionally, there are quite a few pieces missing in order for the utilized development process to cover all aspects considered important in one of the most commonly cited definitions of user centered design.
# Contents

1 Introduction 1

2 Background 5
   2.1 Scrum 5
      2.1.1 Roles 5
      2.1.2 Events 7
      2.1.3 Artifacts 10
   2.2 User experience design 11
      2.2.1 Strategy plane 11
      2.2.2 Scope plane 13
      2.2.3 Structure plane 14
      2.2.4 Skeleton plane 14
      2.2.5 Surface plane 15
      2.2.6 Relationships between planes 16

3 Related work 17
   3.1 Staggered sprints 18
      3.1.1 New role: The user experience team 19
      3.1.2 New event: Sprint 0 19
      3.1.3 Changes to the sprint event 19
      3.1.4 New artifact: User experience backlog 21
      3.1.5 Changes to the sprint review 21
   3.2 Lean UX 21
      3.2.1 Changes to roles 22
      3.2.2 Changes to sprints and sprint planning, themes 22
      3.2.3 New event: User validation session 22
      3.2.4 Changes to the increment 22
   3.3 Effect map 23
3.3.1 Effect map as a product backlog .................................. 24

4 Method ...................................................... 25

5 Result ....................................................... 27
  5.1 Changes to the development team ................................. 27
  5.2 Changes to artifacts ......................................... 27
    5.2.1 Changes to the product backlog ......................... 27
    5.2.2 Changes to the increment ............................... 28
  5.3 Changes to events ........................................ 28
    5.3.1 New event: Sprint 0 .................................. 28
    5.3.2 Generic changes for sprints ............................ 30
    5.3.3 Changes to sprint planning ............................. 30
    5.3.4 Changes to the daily scrum ............................ 31
    5.3.5 Changes to the sprint review .......................... 31
    5.3.6 Changes to the sprint retrospective .................... 31
  5.4 Inspection and adaptation .................................. 31
    5.4.1 Summary from sprint 0 ................................ 31
  5.5 Summary from sprint 1 ..................................... 32
    5.5.1 Process changes for sprint 2 .......................... 33
  5.6 Summary from sprint 2 ..................................... 33
    5.6.1 Process changes for sprint 3 .......................... 33
  5.7 Summary for sprint 3 ...................................... 34

6 Discussion .................................................. 35
  6.1 Fit against Garrets UED model ................................. 35
    6.1.1 Strategy plane ....................................... 36
    6.1.2 Scope plane .......................................... 36
    6.1.3 Structure plane ....................................... 37
    6.1.4 Skeleton plane ....................................... 37
    6.1.5 Surface plane ........................................ 38
    6.1.6 Relationships between planes ......................... 38

7 Limitations .................................................. 39

8 Conclusions and future work .................................... 41

9 Acknowledgements ............................................. 43
References 45
Chapter 1

Introduction

The modern world is constantly changing. Software usage is common in many organizations and companies and as the world changes, so does the needs of organizations and the requirements on the software they use. Organizations that aim at practicing structured software development are in need of a development methodology. A subset of them are considered to be agile software development methodologies, which strive to meet the ever changing requirements in software development projects. There are different agile methodologies available for organizations to adapt. They differ from each other but they all favor the same values [1]:

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

While they acknowledge the values to the right, agile methodologies focus on the values to the left. Adopting agile methodologies often helps organizations to build good software, but they often overlook the design of the user interface [2]. A commonly used strategy for the design of user interfaces is user centered design and the philosophy of agile development might work very well together with user centered design [3]. Theories on how usability ideologies may be incorporated into various agile methodologies can be found in various literature [4, 5, 6].

It is often desirable for a user interface to provide a good user experience. It is difficult to provide a universal model of user experience and the design of the user experience [7]. A frequently cited definition of user experience design is the conceptual model of user centered design proposed by Garret [8]. According to him the design of user experience can be separated into the following five planes:
Surface plane - actual elements of the interface that the users see as they interact with the software such as images, texts, buttons etc.

Skeleton plane - placement of elements in the user interface.

Structure plane - different views of the interface and how the user navigates between them.

Scope plane - features and functionality that the software aims to supply to its users.

Strategy plane - suppliers goals in providing the software and the users needs which the software aim to fulfill.

The planes are ordered from abstract to concrete, where the strategy plane is the first and most abstract plane. As each plane depends on decisions done in the previous planes the strategy plane provides the basis for the user experience. The decisions made on a plane have to align with decisions on the two adjacent planes. If the planes do not align the software may become difficult to piece together, the development project becomes difficult to manage and the user experience of the software will suffer.

Cinnober is a company that builds trading solutions for exchanges and alternative trading platforms world wide. The company delivers advanced solutions to marketplace actors with extreme demands on business functionality, throughput and latency. Depending on the customers needs, Cinnober can supply trading, clearing, market data and/or surveillance solutions.

As a part of the technology toolkit used at Cinnober an internal web framework has been developed. This framework is used in several large customer projects, primarily within the clearing business area. The design philosophy behind this framework is that GUI development can be driven from the data available on the server, through configuration rather than development.

The company practices agile software development and the development method used by the development teams is based on the scrum development framework. Cinnober would like to compare their current development model with an approach that tackles the problem from the opposite direction. Rather than having the user interface emerge as a result of technology and available data, they would like to explore a software development model driven by UCD.

The aim of this thesis is to identify suitable methods for performing user centered design in an agile software development process and to utilize and evaluate those methods in a development project at Cinnober.

The structure of the report will be as follows. Chapter 2 provides a background on scrum as an agile software development method and an abstract definition of user experience design. Chapter 3 presents some of the issues that may arise when merging agile software development with user centered design, and various approaches that one may use to merge user centered design with agile software development methods. Chapter 4 describes the method used in
this thesis project. The development process used and evaluated in this thesis work is presented in chapter 5 along with the thoughts gathered at sprint retrospectives and the resulting changes to the development process. Chapter 6 discusses and reflects on the resulting model and the thoughts gathered from the sprint retrospectives. Chapter 7 presents some known limitations, and chapter 8 presents conclusions and potential future work.
Chapter 2

Background

2.1 Scrum

Scrum is an agile software development framework that is based on empirical control theory and focuses on delivering the highest possible product value taking time of development into account. Below is a summary of scrum, based on a definition of scrum by Schwaber and Sutherland [9]. Empirical process control is based on three fundamentals:

Transparency - significant aspects must be visible to those responsible for the outcome and which those aspects are must be defined so that observers share a common understanding of what is being seen.

Inspection - Scrum practitioners must frequently inspect aspect of the process to detect undesirable variances, but not so frequent that inspection gets in the way of the work.

Adaptation - if an aspect of the process deviate outside acceptable limits that aspect must be adjusted as soon as possible to minimize further deviation.

Furthermore scrum revolves around three aspects: roles, artifacts and events.

2.1.1 Roles

There are three roles for participants in the scrum process: the product owner, the scrum master and the development team, together they make up the scrum team.

To optimize flexibility, creativity and productivity the scrum team should be cross-functional and self-organizing, meaning that they possess all competencies needed to perform their work and they themselves choose how to best accomplish their work.
Work is done iteratively and incrementally with delivery after each iteration. Frequent deliveries gives more opportunities for feedback and the risk of doing the wrong work decreases.

**Product owner**

The product owner is one person but may represent the desires of a group. He or she is responsible for maximizing the value in the work performed by the development team and for managing the product backlog, those wanting to change the product backlog must address the product owner. Product backlog management includes:

- Ensuring clarity in product backlog items for the development team
- Prioritize items in the product backlog to maximize the value of the work the development team performs
- Ensuring transparency in the product backlog to all observers

The product owner may assign the above work to the development team but the product owner still remains accountable. For the product owner to succeed, the entire organization must respect his or her decisions. No one is allowed to tell the development team to work from a different set of requirements, and the development team isn’t allowed to act on what anyone else says.

**Scrum master**

The scrum master has a servant-leader type of role in the scrum team making sure that the scrum team understands and acts according to the theories, practices, and rules of scrum as to maximize the value created by the scrum team. The scrum master also works with improving how outsiders interact with the scrum team. The scrum master helps the team with the following:

- Finding techniques for effective product backlog management
- Understanding the need for clear and concise product backlog items
- Understanding product planning in an empirical environment
- Ensuring the product owner knows how to arrange the product backlog to maximize value
- Understanding and practicing agility
- Facilitating scrum events as requested or needed
- Coaching the development team in self-organization and cross-functionality
- Removing impediments to the development teams progress
– Coaching in organizational environments in which scrum is not yet fully adopted and understood

Additionally he or she works to improve interactions with those outside the team by:

– Leading and coaching the organization in adopting scrum
– Planning scrum implementations within the organization
– Helping employees and stakeholders understand and enact scrum and empirical product development
– Working with other scrum masters to increase the effectiveness of the application of scrum in the organization.

Development team

The development team is a group of professionals. Together they, and they alone, work with developing the product. The development of the product progresses in increments. The development team decides on the definition of done for an increment and strives for the increment to be complete at the end of each sprint. The organization should empower the development team to be as efficient and effective as possible and as the team is self-organizing no one is to tell them how to implement the product backlog.

The development team is cross-functional, its members possess all the skills required to implement the product backlog. Individual members may have specialized skills but there are no titles other than developer. The members of the development team are not to be distinguished by titles or groups regardless of the work being performed, accountability always belongs to the development team as a whole.

A smaller team can more easily adapt but the team has to be large enough to complete a significant amount of work within a sprint. Fewer than three development team members decrease interactions and results in smaller productivity gains. Smaller development teams may have issues with cross-functionality, making it difficult to implement the product backlog. Larger teams require more coordination, nine members is considered to be the upper limit. Larger teams also makes the empirical process more complex and potentially impossible to manage.

2.1.2 Events

Scrum defines a set of events in which the people filling the previously described roles act. All events have a maximum duration but may end early if the team considers the events purpose to be achieved. The events are used to create regularity and to minimize the need for non-Scrum events. Scrum events are designed to be opportunities to inspect and adapt something, e.g. roles, events or artifacts.
The sprint

Sprints differ from other events as all other events occur within a sprint and once a sprint ends a new one begins. A Sprint may not end early but it may be canceled by the product owner if the sprint goal becomes obsolete. A Sprint is preferably one month or less and the sprint duration is consistent throughout the development effort. Sprints should be long enough for the development team to create a significant amount of work but short enough to reduce the risk of wasting time on building the wrong thing. Longer sprints also makes empirical process control more complex.

A Sprint’s scope may be clarified and re-negotiated between the product owner and development team as more is learned but no changes are to be made that would endanger the sprint goal.

Apart from the development work a sprint contains the sprint planning, daily scrums, the sprint review, and the sprint retrospective.

Sprint planning

The sprint planning is done at the beginning of a sprint. The entire scrum team participates and together they decide on the sprint goal, i.e. what to deliver at the end of the sprint, and how they will work to reach the sprint goal.

The scrum master makes sure that the sprint planning takes place, that attendants understand its purpose and that the event is kept within the time limit. The scrum team collaboratively sets the sprint goal, the product owner works to maximize customer value in the sprint goal. Based on the sprint goal the development team creates the sprint backlog, they may negotiate its content with the product owner to optimize the work performed in the upcoming sprint.

The input to this event is the product backlog, the latest product increment and a projection of the development teams capacity based on their performance in previous sprints, and the output is a sprint backlog.

Daily scrum

The daily scrum is a 15 minute event that occurs daily at the same time and the same place. This event lets the development team synchronize and plan for the next 24 hours. They inspect the work since the last daily scrum and forecast what could be done before the next one. They do this by having each development team member answer three questions:

- What did I do yesterday that helped the development team meet the sprint goal?
- What will I do today to help the development team meet the sprint goal?
- Do I see any impediment that prevents me or the development team from meeting the sprint goal?
The development team uses the daily scrum to inspect progress in the sprint backlog and to ensure they are still heading toward the sprint goal, if things are not going as expected they may adapt.

The scrum master ensures that the meeting occurs, that it is kept within the time limit and that only members of the development team participate, but the development team is responsible for conducting the daily scrum.

Sprint review

A Sprint review is held at the end of the sprint to inspect the increment and to adapt the product backlog if needed. During the sprint review the scrum team and stakeholders discuss the increment. Based on that discussion, and changes in the product backlog, the budget and the market, the attendees discuss what could be done to optimize delivered value in upcoming sprints. This is an informal meeting intended to elicit feedback and to foster collaboration.

The scrum master ensures that the event takes place, that attendants understand its purpose and that the event is kept within the time limit. The product owner explains which product backlog items have been done and which ones have not. The development team demonstrates the work that it has done and answers questions about the increment. The development team discusses what went well during the sprint, the problems they ran into, and how those problems were solved. The product owner discusses the product backlog and projects likely completion dates based on current progress. The entire group discuss what to do next.

The input to this event is the product backlog and the latest product increment. The output is a revised product backlog.

Sprint retrospective

The sprint retrospective occurs after the sprint review and is an opportunity for inspection and adaptation.

The purpose of the sprint retrospective is for the scrum team to inspect how the last sprint succeeded with regards to people, relationships, process, and tools and identifies what works and what does not. Based on the evaluation the scrum team create a plan on how to improve the process to make it more effective and enjoyable, and how to adapt their definition of done to improve product quality. Although improvements may be implemented at any time, the sprint retrospective provides a formal opportunity to focus on inspection and adaptation.

The scrum master ensures that the event takes place, that attendants understand its purpose and that the event is kept within the time limit. The scrum master participates as a peer team member in the meeting from the accountability over the scrum process.
2.1.3 Artifacts

Scrum artifacts represent work or value and aim to provide transparency and opportunities for inspection and adaptation. Artifact transparency is important as decisions to optimize value and control risk are made based on the perceived state of the artifacts. The scrum master works to ensure transparent artifacts.

Product backlog

The product backlog is an ordered list of desired features for the product, the single source of requirements for the product. The product backlog is never complete and it exists as long as the product exists. It is a living artifact that evolves as the product evolves and as the surrounding world changes.

Product backlog items have an estimated time and value, they are used together to decide on the order for items in the product backlog. Higher ordered product backlog items tend to be clearer and more detailed than lower ordered ones as that results in more precise estimates. Items at the top of the product backlog also have to be more detailed so that the development team can use them to create a sprint backlog.

The product owner is responsible for the product backlog and its availability and may assess progress towards potential goals as to improve transparency for the stakeholders. The product owner and the development team continuously work with refinement of product backlog items by improving detail and estimates and by adjusting product backlog ordering. The development team decides on the estimates but the product owner may assist them in this. How and when refinement is done is up to the scrum team.

Sprint backlog

The sprint backlog is a set of items from the product backlog, a forecast by the development team about what functionality has to be in the next increment to meet the sprint goal, and a plan for the work needed to deliver that functionality. The sprint backlog is a highly visible, real-time picture of the work that the development team plans to accomplish during the sprint.

The sprint backlog belongs to the development team and they modify it throughout the sprint. As new work is required, the development team adds it to the sprint backlog, and when work is deemed unnecessary it is removed. The development team tracks the total work remaining at least for every daily scrum to project the likelihood of achieving the sprint goal.

Increment

The increment is the sum of all the product backlog items completed during a sprint and the value of the increments of all previous sprints. At the end of a sprint, the new increment must meet the scrum teams definition of done and it
should be in a useable condition. As it must be in usable condition the product owner may choose to release the product after a completed sprint.

2.2 User experience design

There are different approaches for shaping the user experience of a software product or service. User experience design (UED) is one approach, and the abstract model of UED created by Garret [8] will be considered in this report as it is widely cited and easy to grasp.

Garret’s model of UED covers the shape of the entire user experience, to ensure that no aspect of it happens without the designers conscious, explicit intent. In order to achieve this UED is separated into five planes. The surface plane is the top-most dealing with the appearance of images, texts and other elements that users see and interact with. The strategy plane is the bottom-most plane, setting the foundation for the user experience by dealing with user needs and supplier requirements.

The next five sections will go into further details about what makes up each plane.

![Fig 2.1: The five planes of UED, where the strategy plane sets the foundation and the surface plane contains the elements visible for the user [8].](image)

2.2.1 Strategy plane

The strategy plane is made up of decisions regarding two things: Needs of the users and the objectives of the supplying organization. As this plane sets the foundation for all of the other planes, it is important for decisions to be as explicit as possible. Vague decisions on this plane make it difficult to align choices made on the planes above.
All project participants, designers, developers, project managers, need the strategy document to make informed decisions about their work. Other roles within the organization may be beneficial to include as well, such as marketing who’s work will affect the user experience, or customer support whom may have good insight in user needs.

**Supplier objectives**

The supplier objectives can be defined as business goals, the goals that the organization aims to accomplish with the product. The business goals should be specific enough to define the conditions for success but general enough to not define the path to get there. It is important to avoid jumping ahead to identify solutions before the problems are fully understood.

Brand identity also plays a part of supplier objectives as the product brand is more than just a logotype, colors and font faces. People will inevitably create conceptual associations and have emotional reactions of the brand, that can happen by accident or as a result of conscious choices made by designing the product. Decisions regarding the desired perception of the product falls into the strategy plane.

Finally, success metrics should be defined so that progress towards the supplier objectives can be measured. This also makes it possible to measure the value gained from user experience efforts. The success metrics should be trackable, either indirect as in number of calls to customer support, or direct as in the amount of time spent using a feature in the product.

**User needs**

The second half of the strategy plane is focused on user needs. The product is not to be designed for the supplier; it is designed for other people, and if those other people are going to like and use the product, it is fundamental to understand who they are and what they need. Identifying user needs can be complicated, below is a set of concepts and activities that can assist in exploring and defining the user needs a product may help to satisfy.

User needs can broken down into manageable chunks through user segmentation by dividing the users into smaller segments based on their domain knowledge, and demographic or psychographics characteristics. The needs of one user segment may differ from, or even collide with the needs of other segments. User research is a way to develop a sense of who the users are. Different research tools are suited for gathering different kinds of information.

Market research methods like surveys and focus groups can also provide general information about your users, these methods provide more value when you can be more specific about what kind of information you are looking for.

There is a large set of methods that is usually referred to as contextual inquiry. Together they form a powerful toolkit for gaining an understanding about your users and the context they are in. Contextual inquiry is often
time consuming and is best utilized when a deep understanding of the users is required.

User testing can be used to validate the user experience efforts by testing user interactions with the product. This can be done using the finished product or earlier on in the process using prototypes.

Personas are fictional characters created with the data collected from various research. Using personas to represent data helps to keep the users in mind during the design process.

2.2.2 Scope plane

Decisions regarding which features and functions to include in the product belongs to the scope plane. Making these decisions help to find colliding features, creates a common language and sets a reference point for upcoming work.

Clearly stating what is to be built makes it easier to know when the goal has been reached for everyone involved. Knowing what not to build is also important, some features do not fit with the strategy and some are better off being built in the future.

Requirements provide rules for what is being built. Requirements regarding branding or platform support deal with the product as a whole while other requirements only deal with specific features. Users can be a great source of requirements, but often requirements originate from stakeholders within the organization. Regardless of their origin, requirements can be generalized into three categories.

Foremost are the things people say that they want, some of these may be good ideas and can help to improve the value of the product. On the other hand, people may occasionally want things that they don’t actually need or that are impossible to implement. Delving further into these needs may however prove valuable as there may be underlying problems, problems which can be properly solved. Finally, there may exist some features that users don’t know that they need. Getting people to talk about their needs or objectives might help to uncover these new needs.

No matter the target platform, hardware requirements will come into play. Sensors, displays and other means of interaction will affect the possibilities of the product.

Personas can come into play again in the scope plane by describing how a persona may go about fulfilling their needs as simple and narrative stories called scenarios. Creating scenarios may help to find additional requirements regarding user needs.

On this plane, clear definitions are more important than heavy documentation. Strive for clarity and accuracy rather than large volumes. Stay positive, be specific and avoid subjective language.

Collecting requirements can be easy, what is more difficult is deciding on what fits into the scope of the project. Requirements should align with the decisions made on the strategy plane. In some cases, requirements align with
multiple strategic objectives and sometimes it is the other way around. There may also be constraints between requirements, some requirements may conflict, trade-off between requirements may sometimes occur, and in some cases one requirement may depend on another.

### 2.2.3 Structure plane

The structure plane deals with matters regarding error handling, language and structuring of functionality and content. Work done on the structure plane is generally considered interaction design or information architecture.

Based on the users’ impression of the product they will have expectations on its workings and how they may interact with it. The users conceptual models of the product features decides if they will think of a feature as a place or an object, and this will affect how they expect to interact with the product. Knowing the conceptual models users have of the features in the product will help in creating a consistent and intuitive user experience.

There is generally three ways of handling user errors. First and foremost is error prevention, by preventing errors from ever occurring users can avoid frustration and save time. If prevention is not possible automatically correcting might be the next best thing, care should be taken as wrongly correcting an error might harm more than help. If that fails, help the user recover from their errors.

Organizing product content in a such a way that it helps users process it on a cognitive level will improve users efficiency. By separating information into informational nodes helps to control the level of detail, avoid implementation specific solutions such as pages or components and allows the use of common concepts for structuring information. Hierarchical, sequential or matrix structures are some ways of organizing information which can be applied when content is considered as nodes of information.

### 2.2.4 Skeleton plane

This plane focuses on the skeleton of the user interface: the placement of buttons, controls, images, and blocks of text. The skeleton is designed to optimize the arrangement of these elements for maximum effect and efficiency, so that commonly used functionality is easily accessible.

Interface design shapes the skeleton of the product, by selecting the right interface elements to realize a feature and by arranging the elements in a way that is easy to understand and use.

While the structural nodes which features are spread out across is decided on the structure plane, the controls used to navigate between those nodes are shaped on the skeleton plane with navigation design. The goal of navigation design is to provide users with means to navigate between nodes and to clearly communicate the relationship between navigational elements, where the user
2.2. User experience design

Currently is, to which other nodes they may navigate and how they can navigate to them.

Information design extends upon the previous two methods, by focusing on how to present information in a way that match how users think and what they strive to achieve. Information design may also stretch into the two adjacent planes by touching on organization of information and the underlying concept of icons and images.

Wireframing can be a way to integrate the three previously described methods with sketches and notes. Wireframes can be used to describe the arrangement of elements, how navigation is performed and how information should be presented. Gathering all this information into a single document helps to create an overview, which can be used to validate that the decisions made on the skeleton plane align with the planes below and can be used as a reference point during implementation.

2.2.5 Surface plane

The surface plane is about sensory design, focus is now on the presentation of the elements decided upon earlier. Focus is often on visual design but the other senses such as hearing, touch or smell may also be utilized when shaping the user experience.

Visual design is not only about aesthetics, the visuals should be aligned with decisions made on the lower planes, such as branding strategies or navigational matters. Utilize the products visual appearance to help guide the users eyes to what is important and maintain uniformity across the entire experience to avoid confusing or overwhelming the user.
2.2.6 Relationships between planes

When progressing from the strategy plane towards the surface plane the issues that must be dealt with become less abstract and more concrete. Each plane depends on the planes below it, available choices on each plane are constrained by the decisions made about issues on the planes below. This means that decisions made on the strategy plane will ripple all the way up the chain.

This does not have to mean that every decision about a lower plane must be made before the plane above it can be addressed. Dependencies run in both directions, and a desirable decision on a plane may sometimes require a reevaluation of issues on lower planes. Considering decisions made on lower planes as set in stone before work on higher planes have started can therefore cause problems.

Figure 2.2: Work should begin on the next plane before it is considered done on the current plane [8].
Chapter 3

Related work

Some studies show that agile software development is not sufficient when it comes to creating a good user experience[3] and that there may be some difficulties in integrating agile software development with UED. Sy[10] notes that the gathering of user feedback performed by their agile team did not suffice as user data required for their design process. Agile software development tend to focus on internal values, such as team communication and developer responsibilities while a UED process would rather focus on the people using the software than those creating it [3]. As some UED activities are creative or even artistic the path to completion is not clear beforehand and may therefore be difficult to estimate [11].

When breaking UED activities down into smaller pieces so that they may fit into an agile sprint it may be difficult to maintain a holistic view of the user experience. But it is a skill that may be acquired through practice [10].

Some report success in performing smaller UED activities just prior to the implementation of a feature, longer running UED activities still remain difficult to fit into the process of agile software development [11].

The difficulties might be worth the struggle as there are also advantages in integrating UED and agile software development. Sy[10] has found that the integration produces better results than a waterfall approach with the same UED techniques. More of the product was designed, there was a clearer focus on important designs and results were more in line with strategic decisions. UED may sometimes become slow and resource consuming and taking an agile approach to UED may yield better results in the ever changing environment of software development [3].

Blomkvist considers three approaches when integrating agile software development and UED. In the first approach agile software development is used as a base and remains as the foundation while UED methods and values are applied on top of it to some extent. The second approach is the inverse, UED is used as a base while agile methods and values are applied on top. The third is a balanced integration, a difficult task with the aim to create a pro-
cess with methods and values that are both user-centric and agile. The third approach is considered to yield the best result, while the first approach can be advantageous for organizations that are already familiar with agile software development. The first approach might not yield the best results in regards to user experience but can help organizations improve the user experience in what they deliver without having to completely abandon their current approach [3].

Blomkvist suggests further study of various integration approaches and the development process that they result in, which will be the focus of the remaining parts of this thesis[3]. The first approach will be considered as the user experience initiative is recently started at Cimnober and the process of agile software development is already established, the first approach should have a higher chance of being adopted. Without delving too far into Nielsens model of UX maturity[12, 13], this may also fit better with the Cimnober UX maturity grade. The first approach may help to improve the UX maturity to a point where the third approach would be possible to introduce.

There are two outstanding attempts at integrating agile software development and UED: Staggered sprints[10] and the scrum variant of lean UX[4]. The effect map[14] has been another major influential piece although it is not an attempt at integrating agile software development and UED.

### 3.1 Staggered sprints

Sy[10] has developed a process model developed to ensure that products are still designed with users in mind as the developing organization adopted agile software development. This model has later become known as the staggered sprints model[4], and this is how it will be referred to from here on.

The key piece of this model is ”just in time design”, which revolves around the following:

- As developers implement a few features at a time the user experience team can focus on performing usability activities for only those features
- Collaboration between the user experience team and the development team to ensure that design implementation does not drift away from the validated design intent
- As the increment has to be in a done state at the end of each sprint there should not be any incomplete features impairing the user experience
- Contextual inquiry is done continuously throughout the development effort, bringing fresh data to the sprint planning

The user experience team works in a ”separate track” to stay ahead of the development team. That way they can iterate and test designs before they are to be implemented [10].

This model would result in the following changes to scrum:
3.1. Staggered sprints

3.1.1 New role: The user experience team

The user experience team works to improve the end user experience of the product. They do this by gathering data, prototyping designs based on gathered data, and by evaluating prototypes as well as the increment. Their work is not limited to the increment of the current sprint as they also work to prepare for future sprints and with evaluation of the increment from the previous sprint [10].

In order to fit design activities into Scrum the user experience team utilized what is called ”design chunking”, they break design work apart into sprint sized chunks. A design chunk can be completed within the time frame of a sprint and incrementally adds to the overall design. For design chunking to work it is important to have well defined design goals and a clear high level design intent [10].

3.1.2 New event: Sprint 0

Sprint 0 is used to gather requirements. Sprint 0 occurs at the start of the project, before the first sprint. Usability investigation activities in sprint 0 depend on whether the product is the next release of an existing product or completely new. The following activities may be included: [10]

– Gathering data to refine or hone product- and release-level goals. Facilitating the alignment of all team members understanding of these goals, so they constitute a shared vision.

– (For a completely new product) Interviewing or conducting contextual inquiry during customer site visits for market validation, and preparing high-level exploratory designs for market validation. Create design principles that inform and guide design decisions for the product based on gathered data.

– (For an ongoing release) Analyzing and summarizing prior contextual inquiry and usability test data. Based on that data, clarify release-level design goals to inform and guide design decisions through all iterations.

– (For a completely new market or capability) Developing brief and vivid descriptions of target users and workflows (light personas and scenarios) from investigations.

3.1.3 Changes to the sprint event

During sprints the user experience team perform usability investigation activities. Their work is not limited to the current increment as their work also includes evaluation of the previous increment and preparation for the two upcoming increments. In the current sprint, sprint k, the user experience team works with the following:
Designing prototypes for sprint \( k+1 \).

Present designs and gathered data to the development team relevant to this sprint.

Close interaction with the development team as they implement designs in the current sprint.

Conducting contextual inquiry and interviews to investigate designs for sprint \( k+2 \) and to evaluate the prototypes designed for sprint \( k+1 \).

Usability testing the increment from sprint \( k-1 \).

Figure 3.1: The user experience team prepares the two upcoming sprints and validates the previous sprint [10].

During the first few sprints, to give interaction designers time to do these usability investigations, developers should focus on developing features that requires little or no design of the user interface [10].

Similar to the development team, the work performed by the user experience team is iterative and incremental. Design work is split into pieces that can be performed within a sprint, called design chunks. A design chunk builds on earlier design chunks and incrementally add to the overall design. It is not just design and prototypes that incrementally increase in fidelity to the final product, test participants and test protocols do as well. Test participants in early sprints may be coworkers with some characteristics in common with end users while test participants in late sprints are actual end users that commit to a series of tests. Tests in early sprints focus on operation level tasks in artificial environments that may not occur in the real world, while late sprint tests focus on entire workflows and are performed in the end user’s actual environment. By performing early tests with coworkers, the user experience team is able to refine the testing protocols before moving on to testing with end users [10].
3.1.4 New artifact: User experience backlog

The user experience team keeps track of their work in a separate backlog, preferably on a board in the same space as the sprint backlog. The user experience backlog contain issue cards that move across three columns. Issue cards start out with information about usability issues, bugs or feature requests in the usability issues column. As members of the user experience team begin designing a solution for an issue they move the corresponding card from the first column, the ”usability issue column”, to the second column, named ”fix in prototype”. Once the user experience team has found a set of design changes that solve the problem the card moves to the last column, the ”fixed & works” column [10]. When a design is completed user experience team members collaborate with the developers to create items for the product backlog or the sprint backlog that represent the design. An issue card may result in multiple backlog items and for larger issues the user experience team assist developers in splitting the design into chunks that can be completed over several sprints. The user experience backlog is also used to keep track of features requested from design partners, they use two columns for this: one for feature requests that need to be further discussed and another column for feature requests that are deemed ready for the next sprint [10].

3.1.5 Changes to the sprint review

The user experience team may use the sprint review to test the increment with end users at an external site. As they return they present their findings to the scrum team. The presentation is done verbally, supplemented by demonstration of prototypes or the increment where necessary, and may include feedback and usage observations for prototypes and the increment, data on user context and workflows, feature requests, bugs and usability issues. In this way the user experience team is able to replace personas with people, and scenarios with workflows and sample work files. The user experience team also creates issue cards representing most of the presented data, but based on discussions following the presentation some issues may directly result in backlog items [10].

3.2 Lean UX

Lean UX is a deeply collaborative design process based on *The lean startup*[15]. The following five concepts are fundamental in lean UX[4]:

- Reducing document handoffs
- Collaboration between disciplines
- An experimentation based mindset
- Acting on direct observations of peoples needs, wants, likes and dislikes
The four core principles of agile software development

Small, co-located and cross-functional teams where everyone is considered equally important helps to reduce unnecessary work and to gain a wide set of a insight into problems. The team should focus on experimentation rather than analysis and strive achieve outcomes rather than delivering features. While lean UX can be employed on its own, there are also suggestions on how lean UX principles and activities can be integrated into scrum. The scrum variant of lean UX is based on staggered sprints and would impose the following changes [4].

3.2.1 Changes to roles
For the team to be cross-functional all roles should participate in the development process. The insights of the product owner, development team, designers and eventual domain experts should be considered equally important and all of them should be involved in the design process.

3.2.2 Changes to sprints and sprint planning, themes
Sprints are grouped into themes. The theme starts off with an extended planning session which can greatly vary in length depending on the scope of the theme, the extended planning could take a couple of hours or up to a week to complete. During this time the team employs brainstorming exercises and other ideation activities to discover and decide on what is going to be tested and validated in this theme. Collaboration is important and the team may choose to include others to improve the outcome of this extended sprint planning. As more insight is gained during each sprint, each upcoming sprint planning within the theme provides an opportunity to validate the ideas generated within this theme.

3.2.3 New event: User validation session
Validation sessions with users should be planned once per week. This will provide frequent opportunities to check if the team is on the right track. In early sprints focus should be to validate that there is value in the product, while later sprints with a refined increment is more about validating the usability.

3.2.4 Changes to the increment
With an experimental focus, things are more likely to be removed from the increment than what many are used to. Creating minimum viable products is a key aspect in lean UX, doing the least amount of work in order to discard or validate ideas helps to faster reach desired outcomes. It is likely that the increment will contain relatively less code that traditionally. Sketches, prototypes, wireframes or anything that helps to validate ideas should also be considered
3.3 Effect map

In an attempt to avoid insufficient product descriptions the people at InUse\textsuperscript{1} came up the concept of the effect map. The effect map is a way to gather the desired effects of a product together with ideas on which actions should be performed in order how to get there. The desired effect can be expressed on three levels: The ultimate goals in creating the product, who are the people that will help to achieve those goals and what will they do or need (called usage goals) in order to reach the goals. The items on each level and the actions required to reach the desired effects can be organized into a tree hierarchy. Actions are connected to the usage goals they help to fulfill, people are connected with their usage goals and the ultimate goals they’ll help to achieve. All of this can then be prioritized, starting with the goals and then by moving down the tree, eventually creating an ordered list of which actions should be performed in order to reach the most important goal. Measurement points can be set up for goals and usage goals in order to measure the effectiveness of performed actions [14].

\textsuperscript{1}InUse is an agency that assists clients in creating digital products and services, an agency that often share their findings in publications and on conferences.
3.3.1 Effect map as a product backlog

Some choose to describe backlog items as: As user, I want to perform task in order to achieve some goal, which is similar to what is described in the effect map. The effect map also keeps track of the ultimate goals of the product while making it easier to keep track of users and their goals.
Chapter 4

Method

Scrum can be described as a set of roles, events and artifacts. The initial plan was to describe scrum using those three concepts, to investigate how scrum is performed at Cinnober and describe how it differs from vanilla scrum in terms of roles, events and artifacts. The next step was to find a commonly used model of user centered design, figure out a way to integrate said model with scrum and express the changes that would have to be made in terms of roles, events and artifacts. If some changes to the process were required for this specific project those changes would be described in a similar manner. The resulting process was then to be evaluated in a development project at Cinnober.

However, Cinnober was in the process of reworking their scrum approach and there was little value in taking the old scrum approach into consideration.

Integrating UED with scrum was based on findings made by others in their attempts to do the same, and the resulting development process was then validated against a commonly cited definition of UED.

Figure 4.1: Scrum forms a base for the development process while company culture and the constraints of UED inflict changes on the process.
Chapter 5

Result

A development process with scrum as a base and a mixture of aspects and activities from the methods described in chapter 3 was put together. This process was employed and evaluated during a 8 weeks long development project at Cinnober. In line with the empirical process control used in scrum, inspection and adaptation of the process was exercised along the way. Below is a description of the resulting development process, the reflections gathered during inspection, and how adaptation was done according to those reflections. The description is expressed as changes made to the scrum model described in chapter 2.

5.1 Changes to the development team

User experience designers will be part of the development team, they collaborate with the developers in preparation work, design of the user interface and user validation sessions.

The user experience designers may not be fully dedicated to one project at a time and will not always have the time required to participate in all activities where it would be beneficial for them to participate. In this case the rest of the development team would perform the activity and later validate the result with a designer.

5.2 Changes to artifacts

There is three changes to the artifacts with the aim to improve the user experience in the resulting product.

5.2.1 Changes to the product backlog

The effect map is used as a foundation for the product backlog, with the effect map’s actions becoming the actual items in the backlog. Some already choose
to include users and their needs in their backlog items, but the effect map provides a more structured approach. As it keeps track of product goals, users and their needs in a structured manner, it becomes easier to validate if new suggestions for backlog items actually fit into the increment. Proto-personas, a lightweight variant of personas [4], is used to represent users in the effect map. Proto-personas provides a mean to put a face on gathered data and gives a more detailed description of users than simply stating their role, while still being cheaper to produce than a full fledged, regular persona.

Maintaining a structured flow in the user interface can be difficult when working on a per-item, or per-Sprint basis. The navigation and structure between features being developed in the current sprint will have to fit in with what is going to be developed in future sprints. Some design of structure and flow ahead of time will reduce the risk of having to do major rewrites in order to maintain a good user experience in future sprints.

5.2.2 Changes to the increment

Similarly, it can be difficult to maintain a uniform look and feel when working on a per-item, or per-Sprint basis. A styleguide is a tool that can be used to keep track of appearance, behavior and common elements used in the user interface. As the increment changes the styleguide will have to be updated to mirror alterations and additions of elements in the user interface [4].

5.3 Changes to events

5.3.1 New event: Sprint 0

Before the regular sprints start there is now a sprint 0, this event is used to create a common perception of the product within the team, and to start working with the artifact changes described in the previous section.

The scrum team formulates general product goals, and validates the goals with relevant parties. Base on that, the team gathers data on potential users. It is important to involve the team as much as possible during this sprint in order to create a shared understanding of the problem at hand, this can help to reduce the amount of documents that have to be produced. Being involved from the start may also improve the teams commitment to building the product.

The data gathered on users is used to construct proto-personas. This is also an activity of the entire scrum team, but may not require the involvement of external parties. More than one proto-persona will be necessary in most cases, attempt to create diverse personas in order to back up all of the user needs that the product aims to fulfill.

Proto-personas deal with user needs on some level, but that is not extensive enough to create a proper base for the product backlog. Once again the scrum team uses the data they have gathered to create a list of all the user needs they aim to fulfill with the product. The needs should be connected to the
proto-persona that they belong to, some needs may be shared among multiple personas.

Next up is to gather all tasks that the users would perform in order to fulfill those needs, some tasks may fulfill several needs while others may require multiple tasks. For the task items to be usable as product backlog items they may require some refinement, how will the product assist users in performing those tasks? Attempt to find relationships between tasks, are some tasks performed simultaneous or sequentially?

Users, needs and tasks should then be prioritized, it is not possible to implement everything at once. Prioritize users first, needs second and tasks last. The needs of the most important user should be prioritized above the needs of the other users, the same thing goes for needs and tasks.

Figure 5.1: Users, needs and tasks written on post-it notes and prioritized.

Once that is complete it is time to design the skeleton and structure of the product. Questions to answer here are: Which views will the product have to contain in order to facilitate the tasks previously decided upon and how it is possible to navigate between those views. Wireframes for some views may also be beneficial.

The final part of sprint 0 is to create the foundation for a styleguide. The wireframes from the previous step may help to find common elements to add to the styleguide.

It is important for the entire scrum team to be involved as much as possible in this step in order to create a sense of ownership and commitment to the increment and backlog work done in sprint 0. It is important to avoid a heavy handoff from the designers to the developers and strive to create something that everyone can stand behind.
Doing design up front is a risk, work done beforehand may become deprecated and will not provide value. Sketches, plans and styleguides created during sprint 0 should be kept as lightweight as possible to reduce risk while still maintaining the alignment between the decisions made across the planes of UED.

Development work such as planning software architecture and researching different technologies may also be beneficial to perform during sprint 0.

5.3.2 Generic changes for sprints

In this case a sprint length of two weeks should suffice to deliver a substantial amount of value each sprint while the risk of missing the target is still low.

As work is begun on an item from the sprint backlog developers and designers in the development team collaborate on creating detailed designs for relevant parts of the user interface. With continuous collaboration the cost of communicating designs can be reduced. The styleguide and the shared understanding created by collaboration allows for using of cheap paper sketches as documentation for the user interface. All issues may be difficult to recognize at first, leaving some space to tweak the user interface after it is implemented may be beneficial. Sprint backlog items should be validated against the effect map as a part of the definition of done.

5.3.3 Changes to sprint planning

As a part of the sprint planning event (or during backlog grooming sessions) the effect map will have to be revisited and any changes to the product backlog will have to be validated according to how those changes fit into, or potentially change the effect map. Revisit each level of the effect map, starting with users, and validate that the items and their prioritization still are correct. Changes to the effect map may also require changes to structure design and to the
5.4. Inspection and adaptation

styleguide. If it is known that user tests will be performed during the upcoming sprint time should be planned for preparation and execution of those as well.

5.3.4 Changes to the daily scrum

As sprint backlog items require collaboration at the start, the intent to start on a new item should be announced ahead of time so that required parties can plan accordingly.

5.3.5 Changes to the sprint review

Sprint review sessions may sometimes include user testing, especially if users are available and the increment is in a state that may benefit from testing. There are various techniques for performing user tests and which techniques to use is decided upon on a case to case basis.

5.3.6 Changes to the sprint retrospective

The sprint retrospective will intentionally be geared at discussing the introduced process changes that are described in this chapter.

5.4 Inspection and adaptation

Based on issues brought up during the retrospective meetings, changes were made to the development process for the second and third sprints. This was done in order to try out some of the other methods presented in chapter 3. Below are summaries of positive and negative results, as well as suggested and implemented changes.

5.4.1 Summary from sprint 0

Many aspects of sprint 0 were found to be positive. The team found that this event helps to create a mutual understanding of the problems at hand and potential solutions, skeleton and structure sketches were an important factor in this. The styleguide proved to be a useful approach to shape the general feeling of the application. To be able to brainstorm around features from a user perspective without considering implementation of said features was an empowering approach. The effect map assisted in creating user stories. Gathering data on users was helpful in finding the correct target group.

While there was plenty of positive feedback for sprint 0 there were also some areas with room for improvement. Gathering data on users was a slow process and consumed more time than planned, this task was dependent on input from people not part of the team which further caused things to slow down. It was difficult to draw a clear line between user needs and tasks in the effect map e.g. should placing orders on the market be considered a need or a task? It
proved difficult to decide on the granularity of the changes made to some of the artifacts, e.g. to which extent should the styleguide and the skeleton/structure sketches be produced. Some of the mandatory features that help fulfill a need may provide more value than some of the non-essential features of a higher prioritized need but the strict prioritization model in the effect map does not take this into account. Sharing the designer with other projects was time consuming. It was not possible to get hold of actual end users, information had to be gathered from second hand sources of which most had a similar background. There is a chance that the information gathered from them may be biased, that along with a complex domain does not make for a solid foundation.

Based on those reflections a set of proposed changes was put together, both for upcoming sprints and future projects. The following things were noted down:

– As user data is difficult to gather, creating and validating concepts before design work starts could reduce potential waste.

– Take a different approach to proto-personas, treat them like cheap prototypes that can be discarded, validated and iterated upon.

– While the different areas of financial technology may differ from each other, chances are greater than usual that the knowledge of users gained in one project may become reusable in another.

– There are many with knowledge of users within the organization, keeping them in the loop may be beneficial for multiple reasons. As the UED initiative at Cinnober is recently started it may be difficult to get the possibility to study end users, but there is still much second hand knowledge that can be gathered within the organization. Validating the resulting applications from this gathered knowledge with those whom it was gathered from may also help to increase their interest in UED.

– When using second hand knowledge, the more people that information is gathered from the less is the chance of having decisions based on biased information.

– There has to be a way to distinguish the between the actual need and related "wants" in the effect map for the strict priority to work.

5.5 Summary from sprint 1

Compared to sprint 0 the first sprint resulted in a greater feeling of progress, the team found it engaging to start creating. Having something to show also improved engagement from various stakeholders within the organization. The review sessions felt successful. On the negative side, using a separate column in the sprint backlog for design reviews proved unnecessary as the implemented
design was reviewed continuously. Doing all design work for a sprint backlog item when it is picked up proved difficult for two reasons: Sprint planning is difficult to do without any prior design to base estimates on and visual design is difficult to fit into the short time frame. Three suggestions for improvement were gathered from sprint 1:

– With a more expressive styleguide the visual design work could be more lightweight and fit better into the design approach used during sprint 1

– Spending more time on research and analysis between sprints could assist in creating estimates during planning and further reduce the amount of visual design required during sprints.

– Remove the review column from the sprint backlog but continue to perform informal design reviews during sprints. Having a specific column for design review in the sprint backlog could still be useful in larger projects.

5.5.1 Process changes for sprint 2

As of the issues encountered during sprint 1, the lean UX approach to integrating UED with scrum was employed in sprint 2. Sketching and ideation tasks were performed at the beginning of the sprint planning session.

5.6 Summary from sprint 2

The progress in the second sprint resulted in positive feedback from the team and from the stakeholders. Performing sketching and ideation tasks before planning made planning easier. On the other hand the team missed some details when design was done before planning. The feedback gathered from stakeholders during the review sessions was difficult fit into the product backlog, as they could not be derived from the effect map. The team was also starting to doubt if the prioritization of the product backlog was realistic. Three suggestions for improvement were gathered from sprint 2:

– Mix the lean UX approach of designing as a part of sprint planning with "just in time"-design where necessary.

– Put more time into sprint 0 to create a more realistic effect map, and by extent, a more realistic foundation for the product backlog.

– If changes are made to the sprint backlog they should be validated against the effect map.

5.6.1 Process changes for sprint 3

Based on the suggested changes from sprint 2, sprint 3 featured a slightly longer design session during the sprint planning mixed with some "just in time"-design performed during the sprint.
5.7 Summary for sprint 3

Progress and engagement from stakeholders continued to improve in sprint 3. Performing design tasks during sprint planning and as a part of sprint work appeared to be a fitting approach. The team struggled with conflicting feedback when testing the application with different people. Also, the designers priority had to be shared with other projects during sprint 3, which was making work difficult. The team felt that the backlog was losing its connection with the proto-personas created during sprint 0. The basal needs of traders are similar thus the attempt at diverse personas felt unnecessary. Only one suggestion for improvement was gathered from sprint 3:

- Better proto-personas (or some other representation of users) may help the team to better handle conflicting feedback from testing sessions with non-users
Chapter 6

Discussion

With a larger development team it may be beneficial to use the same approach for collaborating between developers and designers as the approach used in Sy’s staggered sprints model. Performing design work one sprint ahead of development could improve communication between designers and developers as designers will have more time to put into their designs before sharing them with the developers.

The effect map can be a useful tool for keeping backlog items aligned with decisions regarding the user experience. As items are added to the product backlog they can be validated against the effect map. If the item helps to fulfill one of the usage needs it is added to the product backlog and if it does not the effect map needs to be revised or the item has to be discarded.

Using lean UX outcomes to represent goal nodes in effect map was considered at first, but it proved difficult due to the type of project and the organization involved. The purpose of the project is to affect organizational culture, to improve influence from the UX team in future projects by demonstrating the results of a UX-centered development process. People, needs, tasks and actions that derive from that goal would not fit with the resulting application, but rather the process that caused the resulting application. Cinnober rarely creates products or services on their own, they assist other organizations in doing so. Thus, the outcomes focused on by Cinnober are not ones that involved end users but rather the customer organizations.

6.1 Fit against Garrets UED model

Comparing the process used in the development project with Garret’s definition of user experience design can provide an idea of how well the development process performs in regards to UED. Below is a discussion regarding how well the development process covered the decisions involved on each plane of UED and how well each plane aligned with it’s adjacent planes.
6.1.1 Strategy plane

Decisions that would be regarded as a part of the strategy plane were taken during sprint 0. During sprint 0 the focus was partially on creating a shared understanding of the problem within the team, which is similar to how all project participants should be involved in decisions made on the strategy plane in Garret’s UED model.

Using a styleguide helps to control the brand identity by dealing with the look and feel of interface elements. But it falls short when it comes to dealing with desired associations and emotions of the brand. The effect map could essentially handle the parts of the brand identity where the styleguide fell short, but it would have to be used in a different manner than how it was used in this project.

The granularity of the supplier objectives was not touched upon. The coverage of supplier objectives in sprint 0 would depend on which disciplines that are involved in the project. This project only involved development and design, but other disciplines are often involved in projects at cinnober. E.g. marketing, support and business intelligence may consider other objectives to be important.

The development process used did not include success metrics.

Not jumping ahead to thinking about solutions was difficult at times, according to Garret it is important to fully understand the problem before considering solutions.

User needs

The development process used leaves room for user and market research during sprint 0. The amount of time required for this will differ between projects but the time spent during this project was not long enough. The development process does not provide any method for the actual gathering of user and market data. As users and customers were not available in this project, the team turned to people with good domain knowledge when gathering data on users and the market, and to validate the user experience efforts.

Proto-personas fulfill a similar purpose as personas. While proto-personas and parts of the effect map can be a way to represent user segmentation, a way to gather and process data to understand user segmentation is still missing.

Creating a shared understanding of the problem at hand for everyone is involved is an important aspect of sprint 0, all involved should be aware of the decisions made on the strategy plane.

6.1.2 Scope plane

The effect map proved a useful tool for documenting features and functions that users desire, while keeping track of which needs they help to fulfill. But there was no way to distinguish between features that users actually need and features that they say that they need. The effect map has the potential to deal
with stakeholder requirements, but stakeholders were not included in the effect map in this project.

The development processes failed to discover user constraints with regards to platforms, hardware and interaction possibilities.

The strive for shared understanding creates a clearer view of the goal for the team and helps to keep documentation down. The strict prioritization in the effect map helps in deciding on which features that fit into the scope.

### 6.1.3 Structure plane

Tasks are grouped together based on user behavior during sprint 0, but how this is to be done could be specified in more detail. The conceptual models of the users are not touched upon.

The process lacks a proper way of dealing with the matter of handling user errors. Some efforts were made to deal with user errors after sprint 0 but did not adhere to the model of prevention, correction and recovery. It could probably be beneficial to make some generic decisions on user errors during sprint 0, and more specific decisions during sprint planning.

The structuring of users, needs and actions during sprint 0 had similar characteristics to what garret calls information architecture, but the approach was not as thorough as his model.

### 6.1.4 Skeleton plane

The skeleton was mostly designed during the end of sprint 0, but some skeleton plane type of activities were also performed during regular sprints.

Interface design was performed on an abstract level during sprint 0, based on gathered knowledge the team decided on which type of components and controls would be most suitable for certain user actions, without going into details. As an example: During sprint 0 it was decided that a table with tabs used for filtering would be used for watchlist management, but the exact placement and existence of buttons, labels and columns were not decided upon until sprint 1.

Due to the fairly small scope of this project the navigational design did not have to be as complex as it may have to be in other projects. As it was possible (and necessary) to fit all functionality on one screen navigational design became mostly a matter of placing components in such a way that it would match the users workflows.

It is difficult to discern what is considered interface design or navigational design from what should be considered as information design. The tasks that help cover interface design and navigational design could probably be considered to cover parts of information design as well. It could be beneficial to spend more time and resources for figuring out how users think and what they strive to achieve.
Wireframing and paper prototyping was a commonly used tool for communication between design and development both during sprint 0, sprint planning and throughout regular sprint work.

6.1.5 Surface plane

Visual design was performed both during sprint 0 and the regular sprints. Visual design during sprint 0 involved the creation of a styleguide. It was not used to its full extent but it was still a useful tool for maintaining a uniform look of the user interface. The development process specified when visual design was to be performed during regular sprints, but did not describe how it was to be done.

6.1.6 Relationships between planes

The effect map was a useful tool when it came to aligning the features and functions decided upon in the scope plane with the user needs from the strategy plane. The styleguide helped to align visual design on the surface plane with brand decisions made on the strategy plane.
Chapter 7

Limitations

The results reported here are limited to the experiences of the user experience team within the development environment of a single company, and in a single project which differs from regular projects. The phenomena observed may not extend across all company cultures or to other development projects.

The development team in this project consisted of two people while the suggestion is at least three [9]. It could be argued that this suggested minimum should be increased when the team also consists of designers, as their responsibilities differ from the other members of the development team. As communication becomes more difficult when more people are involved, the process used in this project may not be as useful in a development team of a proper size.

Some assumptions were made when the model of scrum used in integration methods does not fully match with how scrum is defined by Schwaber and Sutherland [9].
Chapter 8

Conclusions and future work

The aim of finding suitable methods for performing UED within an agile software development process was met, utilizing and evaluating those methods in a development project yielded valuable input on the development process. While there was plenty of positive feedback from the development team, they also identified room for improvement in the development process which should be taken into consideration in future projects. There are also quite a few pieces missing in order for the development process to cover all aspects of Garret’s user experience design.

ISO13407 presents human-centered design as a multi-disciplinary activity and thus it is not only usability experts that need to be involved when designing systems in a human-centered manner [16].

Kniberg has shown some evidence of culture affecting the development process. Even if an organization is introduced to new processes that yield great results, it may fail to adapt and maintain such processes because of the culture within the organization [17].

Nielsen presents a maturity model for organizations that practice user-centered design in some way. According to this model organizations move through a sequence of stages as their user centered design process evolves [12].

Based on the three paragraphs above, designing good user experiences does not only rely on the development process but also the culture of the organization and its "UX maturity". While evaluating methods for integrating UED with agile software development is the main focus in this thesis work, finding ways to change the organizations culture to a more UX centered one is also of interest.

Miller notes that it could be fitting for an interaction designer to take on the role as product owner. The interaction designer would have to understand and accept the concepts of agile software development but adds the possibility to keep end-users in mind when creating and prioritizing items in the product.
This could be an interesting approach to try out, the role of product owner may be a more fitting role for an interaction designer rather than being considered a part of the development team.

Blomkvist discussed how automated testing makes development less cumbersome and that automated tests could empower the team to make changes in the user interface. Automated tests are useful to verify functionality that can be tested programmatically, but usability aspects are often difficult to verify in this manner [3]. Could it be possible to perform usability tests in a similarly frequent manner?
Chapter 9

Acknowledgements

Thanks to everyone at Cinnober for sharing their domain knowledge and for the access to the technologies developed by Cinnober, and a special thanks Kajsa Olofsson for helping out to utilize and evaluate the resulting development process. I would also like to thank my mentor at Umeå University, Håkan Gulliksson for always being reassuring and for the support he provided me with when writing this thesis report.
References


