Collaboration between software developers and UI designers in agile software development teams - A literature survey and empirical experiences

Kim Martesuo

Helsinki April 28, 2019

UNIVERSITY OF HELSINKI
MSc Thesis
Master's Programme in Computer Science

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Studieprogram — Study Programme
Faculty of Science	Study Programme in Computer Science

Tekijä — Författare — Author
Kim Martesuo

Työn nimi — Arbetets titel — Title
Collaboration between software developers and UI designers in agile software development teams - A literature survey and empirical experiences

Ohjaajat — Handledare — Supervisors

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
	April 28, 2019	0 pages + 4 appendices

Tiivistelmä — Referat — Abstract

Creating a user interface (UI) is often a part of software development. In the software industry designated UI designers work side by side with the developers in agile software development teams. While agile software processes have been researched, yet there is no general consensus on how UI designers should be integrated with the developing team. The existing research points towards the industry favoring tight collaboration between developers and UI designers by having them work together in the same team. The subject is gathering interest and different ways of integration is happening in the industry.

In this thesis we researched the collaboration between developers and UI designers in agile software development. The goal was to understand the teamwork between the UI designers and developers working in the same agile software teams.

The research was conducted by doing semi-structured theme interviews with UI designers and developers individually. The interviewees were from consulting firms located in the Helsinki metropolitan are in Finland. The subjects reported about a recent project where they worked in an agile software team consisting of UI designers and developers. The data from the interviews was compared to the literature.

Results of the interviews were similar to the findings from the literature for the most part. Finding a suitable process for the teamwork, co-location, good social relations and a an atmosphere of trust were factors present in the literature and the interviews. The importance of good software tools for communicating designs, and developers taking part in the UI designing process stood out from the interviews.

CCS -> Software and its engineering -> Software creation and management -> Collaboration in software development

Avainsanat — Nyckelord — Keywords
design, software development

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — övriga uppgifter — Additional information

Thesis for the Software Systems study track

Contents

1 Introduction **1**

2 Background **3**

2.1 Collaboration in software development teams 3

2.2 User-centered design 4

2.3 Integration between UCD and agile software development methods . . 6

2.4 Collaboration between developers and designers 8

2.5 Summary of the literature review 12

3 Research design **14**

3.1 The interviewee projects 14

3.1.1 Project 1 14

3.1.2 Project 2 15

3.1.3 Project 3 16

4 Results **18**

4.1 Project success 18

4.2 Process flow 19

4.3 Collaboration between the developers and designers 25

4.3.1 Close collaboration compared to separation between developers and designers 25

4.3.2 Communication 27

4.3.3 Big picture understanding 30

4.4 Factors for good collaboration 31

5 Discussion **39**

5.1 Answering th research questions 41

5.1.1 RQ1 41

5.1.2 RQ2 42

5.2 Related work 43
5.3 Validity and limitations 44

6 Conclusion **46**

References **48**

Appendices

A Interview data **0**

1 Introduction

User experience is an important part of software [usa]. This is especially true for applications like web-sites used by the public since user experience affects the sites traffic and potential business opportunities. One possible way to improve the user experience (UX) is to invest in the user interface (UI) and its usability. To do this the software projects might include designated specialists for this task in the software project. These specialists design the user interface of the software to be developed.

In this thesis we will call these specialists with the term designers. The designers produce, or in other words design, ui-designs which are physical or virtual visualisation of user interfaces. The ui-design can be in different forms like electronic or just plain hand drawings.

The role of the designer is not specifically defined in agile software methods like Scrum and Kanban. These methods don't define where the designers position is at all. For example whether it is in the team developing the software or on a higher level of the software project outside the developing team. While there is no general agreement on the designers position the integration of design work and development is happening on the field. How the design practices and agile processes should be integrated is gathering interest [MCA16, BMMW15, SK10]. In this thesis, with the term agile team we mean a team using an agile software method as a base. The method can be Scrum, Kanban or some similar process that is iterative in its nature.

A fair amount of research has been conducted about the integration between user-centered design (UCD) and agile software development methods [MCA16, SK10, CSM06, BMMW15]. The research is mostly about the methods for integration. The existing research gives some insight on the collaboration between designers and developers especially on a higher level. The literature review conducted in this thesis showed that integrating the developers and designers into the same team seemed to be the favoured approach for the integration. This way the two groups work closely together. There is not that much research about the collaboration on the team level and about the daily work between designers and developers. On the other hand there is a growing interest in the people dimension of the integration [BMMW15] and after all the collaboration between the two groups has a lot to do with social relationships.

In this thesis we will be researching the collaboration between developers and de-

signers in agile software development when they are a part of the same team. This is because a way of integration with close collaboration is getting attention. More precisely, the context is co-located agile software teams where the designers and developers work together but have specific job descriptions. Specific job descriptions means that the developers code and the designers design the user interface. Neither does both.

We are interested in the challenges that designers and developers face in their daily teamwork but we are also interested in the practices they find good for their teamwork. We have defined the following research questions:

1. What are the challenges in the collaboration between designers and developers working in agile teams?
2. What practices are useful in the collaboration between designers and developers in agile teams?

In the empirical part of this thesis we will gather data by conducting interviews with developers and designers that have or are working in agile software teams composed of the both groups. We expect to get insights about the teamwork from both parties. The gathered data will be compared to the existing research and we will answer the presented research questions.

The rest of the paper is organized in the following way: section 2 discusses the background research, in section 3 the research design is presented, in section 4 we present the results and in section 5 we discuss the results. Section 6 is a conclusion of this thesis.

2 Background

The material for this section has been gathered using the snowballing technique. The three pieces of work that were used as seeds are Kati Kuusinens "Task Allocation Between UX Specialists and Developers in Agile Software Development Projects" [Kuu15], "From Minimum Viable to Maximum Lovable: Developing a User Experience Strategy Model for Software Startups" [Hok17] and "Focal Points for a More User-Centred Agile Development" [BD16]. First these three articles were read. Suitable references were identified from the texts and reference lists. The titles and abstracts of these references were read and if they were suitable for the subject of this thesis they were chosen. The snowballing was then continued. For some articles we checked articles that referenced them to find more recent articles about this subject.

The articles used as seeds were hinted to me. Other sources my supervisor hinted to me were "Collaborative Software Engineering" [MGHW10], "User-Centered Design and Agile Methods: A Systematic Review" [SMMS11] and "A systematic literature review for agile development processes and user centred design integration" [SPC14].

The designers job is to design the user interface. This means that they design how the user interface should look like on the device the user is using. In practice this means for example where things are placed in the user interface, how different user flows work, how interactions work, how buttons should look like and a lot more. They also design the styling of the user interface (colors, fonts etc.). This work can be done with special software meant for user interface designing or even just with a pen and paper.

The developers are the software developers of the team. They are the programmers that build the software. The developers tasks contain for example designing the technical architecture of the software, writing code, maintaining the infrastructure and implementing the ui-designs that the designers provide.

2.1 Collaboration in software development teams

A team developing software consists of several persons and their job is to create a product through working together. A software team is responsible for the technically creating the product and the team must me formed, organized, managed, evolved and in the end disbanded [WMGvdH10].

Whitehead et al. defines collaborative software engineering with four edicts [WMGvdH10]: *1) "any software project with more than one person is created through a process of collaborative software engineering"*. With multiple persons in a team, the team members must use teamwork to produce the software.

2) "software engineering collaboration is the mediation of the multiple conflicting mental conceptions of the system held by human developers". Members of a software team have different backgrounds and experiences. When collaborating all the personal features of these members come together and form a system.

3) "software engineering collaboration also involves the joint identification and removal of error". Members of a team view systems in different ways. Errors will appear in the system and through collaboration the team identifies and removes errors. In a software team this is relevant mostly for developers that are using methods like code reviews and for testers whose sole purpose is to find errors in the software.

4) "software engineering collaboration is about creating the organizational structures, reward structures, and work breakdown structures that afford effective work towards goal". To have effective team work the members of the team must be motivated to work efficiently together. Proper rewards must be in place and everybody should be aware of their responsibilities. Good leadership supports the collaboration.

Collaboration happens on all levels of a software project. There is collaboration between the customer and the developing team, between the end-user and the customer and developing team, between the members of the developing team and so forth. In this thesis we focus on the collaboration between developers and UI designers in a team.

2.2 User-centered design

The term user-centered design (UCD) is apparent in the gathered background research and is therefore used throughout this thesis.

First we will differentiate between the term customer and user or end-user. The customer is the party whom a certain software is being developed for. In agile software processes the team works to serve the customer. In case if the team is internal to a company the "customer" can be for example a product owner. The end-user or user (terms used interchangeably) is the entity that will use the software that is developed. Figure 1. illustrates the differences.

User-centered design is about involving the end-user in an iterative design process of a computer system [GL85]. The approach has stemmed different techniques that involve the user since its proposal in the mid-1980s by Gould et al. [CSM06, GL85]. In certain parts UCD is quite similar to agile software development processes. Mainly, both are iterative in nature. On the other hand the customer is the central part in agile software processes and the ultimate goal is to produce value to the customer whereas in UCD the end-user has a big role.

In a UCD process the end-user is often brought in to help decide the path for the ui-design through testing. Prototyping is an important part of UCD. In prototyping multiple designs are tested with end-users and the development is then pivoted based on some prototype. The point of this is to achieve good usability and user experience by involving the user throughout the design process [GL85].

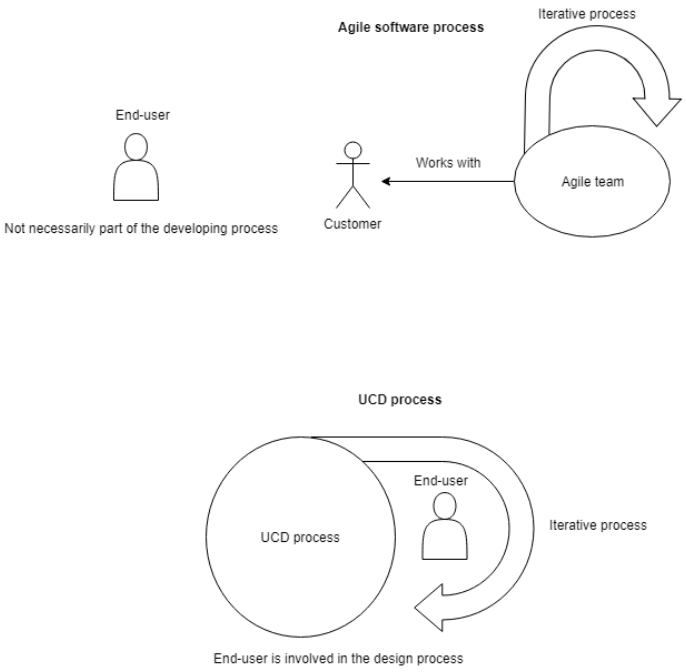

Figure 1: UCD processes vs agile software development processes

2.3 Integration between UCD and agile software development methods

As stated earlier there is a fair amount of research about the integration between UCD and agile software methods. This research is not directly about the topic of this thesis but it can give some insight on the subject.

The integration of these two methods is about how to get the agile development process and the practice of designing to work together.

Integrating user-centered design with agile software development can be done in different ways. For example the design process can be its own track parallel to the development track or it can be interwoven with the development process into one process with more close collaboration. Nevertheless integrating the two is seen as a step forward compared to having the two processes working individually having little to no collaboration [DSMS18].

In their systematic mapping study Magues et al. [MCA16] found different integration proposals for integration between agile processes and usability but no generally applicable guidelines for the integration.

Depending on how you integrate the two processes the designers position in the scope of the project varies. Kuusinen [Kuu15] found three cooperation types for designers in software projects. In other words there are different levels in software projects that the designers can be placed in. The first type is minimal cooperation where one or many designers conducts the design work separately from the rest of the team. From this separate position they provide the projects with designs. The second type is close cooperation between the designers and the project owner. In this type of cooperation the project owner and designers collaborate when producing the designs and provide ready-made designs to the developers. The third type is close cooperation between the designers and the developers. In this type the designers and developers collaborate closely on the UI work.

In Kuusinens research the third type was most favored one. Designers being directly involved in projects is quite common in the field [SMMS11]. Designers and developers are experts in their own fields and these fields are quite different from each other. In this third type, identified by Kuusinen, the designers and developers are working on the same project so it is important that the collaboration between the two groups works well.

Additionally to close collaboration of designers and developers being the favored way

of working, it also improves the collaboration according to some research [Kuu14, Kuu15, SMMS11].

Other research also shows that integrating the designers straight to the agile team seems to be the favored approach for the integration [BMMW15, SK10, MCA16, DSMS18].

For example Sohaib et al. suggests a few approaches (Table 1) when combining principles of usability engineering with agile methods concepts. Sohaib et al. uses the definition for usability engineering by Nielsen [Nie92]. Usability engineering is very similar to UCD. The main principles in both are user testing, prototyping and an iterative design process. They derived the approaches from their literature review[SK10].

Agile Methods Concepts	Usability Engineering	Suggested Approach
Deliver working software frequently	Traditional software approach but iteration within phases	Iterative development throughout the project
Requires generalists	Requires specialists	Assemble a multidisciplinary team to ensure complete expertise
Customer focus	User focus	Collaboration between customers, users, product managers, Business analysts, developers, will maximize overall team efficiency for usable product.
Test driven development and continuous integration	Contextual inquiry, field surveys, usability inspection methods for testing	Unit Testing + User Acceptance Testing + Usability testing throughout the process
Using onsite customer, functional requirement are encapsulated as user stories	Scenario based design for requirement analysis	Integrate user stories with scenario based design

Table 1: Suggested approaches when combining usability engineering (very similar to UCD) with agile methods [SK10]

From table 1 we can see that Sohaib et al. suggests assembling an multidisciplinary team when combining usability engineering and agile methods. They also suggest collaboration between the different parties to maximize efficiency.

Even though Magues et al. do not identify any general guidelines or suggestions they found that several integration strategies integrate the designer role into the agile team directly and this has been met with acceptance [MCA16, BMMW15].

This goes hand in hand with the suggestion from Sohaib.

As we can see from the research done, having the designers and developers work closely in the same team is a favored and accepted way of working and it is seen to be a good practice for achieving better collaboration.

A combined team means that the designers and developers will have to work closely together when delivering software. The daily work in this collaborative environment is a concern for the two groups and it is unclear for them on how to make it work [DSMS18].

What favors the integration is the iterative nature of agile methods and UCD. They both see the user as an important part of the development process (agile methods mainly see the customer as important) and they both emphasize the importance of team coherence. These factors mean that they have the potential to work well together. While they both have similarities they also have differences (see Table 1). For example it is wise that designs are documented somehow and tools are needed to communicate the designs to the developers. Agile methods for one aim towards minimal documentation. A lack of documentation has been reported to cause issues in the integration like confusion and issues in decision making regarding the UI [SPC14]. Several techniques are used in the field for documenting ui-designs and design choices [SPC14], for example the use of wiki pages, personas, prototypes and wire frames. Another way they are different in is that agile methods are against big up-front investigation whereas designers might want to conduct research and requirements mapping before designing the ui-design for example by doing extensive user research [CSM06, SK10, MCA16, BMMW15, JHM14, SPC14]. One way to achieve this is to have a pre-development phase when designers can do pre-work [SPC14].

2.4 Collaboration between developers and designers

The research presented in the earlier subsection discusses ways to integrate design with development. Close collaboration between developers and designers seemed to be the most favored way to integrate the two practices. Next we will discuss what the research says about the collaboration between developers and designers when they are working closely together.

Chamberlain's [CSM06] field study that investigated several teams found four emerging themes from the integration between UCD and agile development. These four

themes were user involvement, collaboration and culture, prototyping and the project life cycle. For the context of this paper the theme "collaboration and culture" is the most interesting one. Collaboration and culture is specifically about collaboration between designers and developers within a team and their culture created by their chosen methodology. Chamberlain's study provides interesting insights for our subject.

As with any teamwork problems stem from the social behaviour of the people working together. For example, one problem is the power struggle that emerges between the groups of designers and developer [CSM06, JHM14]. The power struggle can be seen in that the different groups seem to try guard themselves against having to make decisions at the expense of another group [CSM06]. Power struggles can be countered with the help of proper cultural change and organizational support [BMMW15].

Because development needs ready ui-designs to be able to develop the software it is important that the flow of ready ui-designs is steady. It is clear that the amount of available resources also affect the collaboration. A lack of resources, for example the lack of designers, is reported as a cause for problems in the collaboration [CSM06, BMMW15, JHM14]. Several articles report designers being overworked due to overwhelming workloads and a lack of people resources [Kuu14, BMMW15, JHM14, SPC14]. If there are not enough designers to do the designing work the flow of ready ui-designs is disrupted and the developers might have to wait for ready ui-designs before they can start developing. Especially if the designers are shared between several teams a lot of their time is spent in meetings and the context switching throughout the day can be burdening [SPC14].

Some issues are not directly a cause of the process that the team is working with but rather because of the lack of resources. By fine tuning the process it might be possible to speed up the flow from ui-design to development but a lack of resources is something that can be countered only a certain amount without adding resources. Of course the lack in resources can be overcome with proper resource and project management [CSM06, BMMW15].

Designing and developing are very different tasks. The work require different amounts of time. It is not really efficient to start developing certain parts of the software before the ui-design is outlined. In the design process the designers might ponder between a couple of ui-design options and do user research before selecting the final ui-design. All the details of the ui-design don't necessary have to be decided before

development can begin because small details are easier to change than bigger entities. To make the two practices work well together the design work has to be timed well. Good timing of the design work is also important for project success [Kuu14]. But since designing and developing are very different tasks it is hard to synchronize the two practices [CSM06].

There exists some techniques to try to achieve better synchronised collaboration [BMMW15, SASR16, JHM14, SPC14]. Some techniques try to have the design process ahead of the development [SASR16, BMMW15]. For example by having the design process one sprint ahead of development or using methods like LDUF (little design up front). The idea with having the design process ahead is to design features one sprint before the sprint they are supposed to be developed in. This would supposedly give the designers sufficient time to design a feature and the developers would not have to wait for the designs. In practice it can be hard to keep this flow ongoing. For example if the designers are too overworked they might not just have the time to finish the designs before the developing beings. Staying one sprint ahead can be challenging and in fact designers often find themselves on the same sprint as the developers or behind them [JHM14].

Even though it can be hard for the design process to stay ahead of the development LDUF is seen as an important necessity in agile environments according to research [BMMW15] and it is a common artifact in the research about the integration of UCD and agile software development [SMMS11]. On the other side synchronisation might not be an issue for smaller co-located teams where communication is straight and unobstructed [BD16].

In the case study by Bordin et al. [BD16] the team was small with everyone physically located in the same space. Therefore it was sufficient to handle the synchronisation through direct ad-hoc communication. The working processes were not separated and they all used the same ticketing system for user stories. Interestingly the sprint meetings only involved developers, this reflected the company's overall attitude which was strongly technical. A systematic literature review by Salah et al. concluded that continuous communication should prevail between the developers and designers to avoid delays and bottle necks in the development process [SPC14].

In a combined team of developers and designers the roles can be unclear which can cause problems and confusion. Therefore the designers role with its responsibilities and authorities needs to be clearly defined [BMMW15]. So even though there is no definition for the designers role in agile software methods the team should define the

role clearly themselves to get the teamwork to work well. Bordin et al. supports this by emphasizing the importance of task ownership [BD16]. With task ownership it is meant that everyone knows who is responsible for which task, especially regarding design activities. It should be clear who makes the final decisions regarding design so that technical developers don't implement their own interpretations.

Communication issues like disparity in information between developers and designers seems to be a challenge in the collaboration [CSM06]. But having the developers and designers in the same physical location and providing enough spaces for formal and informal meetings improves the overall communication [BMMW15, BLB12, JHM14, SPC14].

Reaching a shared understanding about the project is a concern [DSMS18, SMMS11]. People in an agile team might not be on the same page of the project which can make the collaboration harder. Developers might just focus on coding while not really having a sufficient image on what the software they are producing really is about. On the other side designers have spent time designing the user experience and have a vision of the product. The design process requires continuous research, design work and evaluation. The results of the design work should be shared to the whole team to achieve a shared understanding of the product. This can be done for example by sharing documents and prototypes and having a sufficient amount of meetings to go through the progress. Having mutual project goals for both designers and developers and an overall big picture of the project throughout its lifetime is important for achieving successful results and a shared understanding between the two groups [Kuu14, DSMS18, SMMS11, SPC14].

Several practices aid in facilitating the collaboration [BS12, SMMS11, SPC14]. The practices can be for example joint designing sessions, the use of sketches, user stories and brainstorming sessions with other team members. Another practice is concept mapping where design scenarios and developers stories are linked together. Most of the collaborative meetings between developers and designers seem to be about so called alignment work. Alignment work is about aligning project objectives with each other [BLB12]. This happens in meetings intended for alignment work and in impromptu meetings between designers and developers. Well facilitated collaboration can lead to transfer of knowledge between groups, achieving a shared understanding, supporting creation and innovation, and team cohesion.

In the end, agile teams consist of several individual persons with different personalities and sets of skills. This means that teams can be quite different from each

other. Different personalities affect the teams ways of work and the teams communication culture. Social relationships play an important part. As long as there is a lack of more established approaches for the collaboration it is still strongly person-dependent [Kuu14]. Barksdale et al. discuss how healthy relationships are important for the team. This can be seen for example in that it is easier to ask for help, get support during difficult times and generally having a more fulfilling work experience in teams with healthy relationships [BS12].

Organizational culture and support can alleviate certain problems (for example communication issues and power struggles) in the collaboration [BMMW15, DSMS18]. Even though designers are a separate role in the agile team the user-centered design integrated with agile development should be seen as a team effort. An understanding organizational culture can help with the integration [DSMS18].

2.5 Summary of the literature review

There is research about integration of agile software development and UCD. The research explores how to integrate the two practices. The two practices have the potential to work well together due to their similarities but they also have differences. Integration types where the designers and developers work closely together is the favored way for the integration.

The research recognizes the following challenges in the close collaboration between the two groups:

- Issues in social relations like power struggles
- A lack of designing resources
- Timing and synchronization issues
- Communication issues

To mitigate the damage from some of these issues, the research presents the following factors.

- Design up front is important
- Good overall big picture of the software for everyone

- Facilitating collaboration through different practices. Helps in mitigating communication issues
- Good social relations
- Organizational culture and support

3 Research design

The empirical part of this thesis was conducted by doing semi-structured theme interviews with both designers and developers. The audio of these interviews were recorded and transcribed. The interviewees answers to presented questions were grouped according to RQ1 and RQ2.

The unit of analysis is a single team member. We are interested in the team members experiences about the collaboration from their point of view. The research subjects are from a few different consulting companies from the Helsinki metropolitan area. Two designers and two developers with experience from projects with close collaboration between designers and developers were chosen for the interviews. Since our goal was to get an understanding of the collaboration between designers and developers in agile teams the only criteria was that the persons have worked in an agile software developing team which has consisted of both developers and designers. To get relevant data we asked the subjects to tell more closely about a recent project that is fresh in their memory but also to talk about the subject on a general level.

With the data we will try to answer the presented research questions. We will try to identify themes and concepts that touch the subject matter. These will then be compared to the findings from the literature review.

3.1 The interviewee projects

Table 2 contains a summary of the reported projects. The interviews were done from the perspective of the interviewees. This means that we did not select projects and interviewed the persons of that project but we selected the persons and asked them to tell about a project. Therefore all team members of the projects were not interviewed.

3.1.1 Project 1

The software developed in Project 1 (P1) is an invoicing application. It is a premium product distributed for free meaning that certain features require a paid subscription. The team in the project develops features which focuses on getting customers converted into premium customers. The team consists of 4-5 developers, one designer and one product owner. The project is continuous and has been going on for five years.

In this project the process from a feature idea to an implemented feature goes basically in the following way: First the product owner checks with the developers if the feature is technically possible. Then the product owner and the designer goes through and agrees on the feature and its specifications. This part does not really include the developers unless some more technical questions arise. When the ui-design is ready for development it is presented to the developers in the next sprint planning. The developers can then comment on the ui-design from a technical point of view where after the designer makes necessary adjustments. Once the ui-design is approved it is taken into the next sprint. The team tends to first implement an MVP of a feature ant bit by bit add more features on top. They also use A/B testing to determine which solutions to further develop.

In the sprint the task is assigned to a one developer or a group of developers. If questions or change requirements appear during the development phase by the developers the designer updates the ui-design accordingly.

Developer 1 (DEV1) is a lead software developer and software architect with a 12 year experience in the software industry. DEV1:s role in the team is developer. DEV1 has been a part of the team for about one year.

3.1.2 Project 2

The software that is developed in Project 2 (P2) is a dashboard for a waste management system. The dashboard contains a lot of visual elements like graphs and different charts. The project is continuous. When Designer 1 (DES1) was a part of the team it consisted of a project lead, one designer and two developers. The team also included a product owner from the customers side (different from the project lead) and another designer who did service design in the beginning of the project for one month.

The project had two week long sprints that were mostly for the developers. The designer had no specified working process. Her way of working was more chaotic. Her only requirement was that she just had to have the designs ready before the sprint started for the developers. When handing over a ui-design to a developer they went through the designs together. The product owner and the designer met once a week for going through the designs and make decisions. The designers conducted user interviews two times a week during the project. The designer and the developers had ad-hoc discussions about any obscurities in the designs. Whenever

a feature was ready during the two week sprint the designer would review the implemented ui-changes together with the developer who developed the feature. The designer also approved the feature before it was considered done. The team had two electronic backlogs. One backlog containing whole feature entities and this backlog was managed by the designer. She also used this backlog to track her work. The second backlog was meant for the developers and their progress, the designer did not use it at all.

Designer 1 (DES1) is a senior UX- and UI-designer with five years of experience in the software industry. She was a part of the team for four months.

3.1.3 Project 3

The software the team is developing is a digital consumer service with over a million registered users and hundreds of thousands active users. The team is a part of a larger gestalt of several teams working on the same service but for different platforms. The team consists of one developer, one dedicated designer (designers from other teams might chip in for the web team), one product owner and one scrum master. The team size has changed and there used to be two developers.

Most of the ideas for features come from the customer but some of them arise after end-user testing which the designer conducts twice a month. Also some ideas arise from feedback by users using the version of the software that is in production. The process from getting a feature implemented starts with the designer, designers from other teams and the customer together first planning the feature together. The designers have their own backlog and they try to stay ahead of the developers. Whereas developers have their own backlog for their development tasks. Once a feature is specified specs and the ui-designs are ready it can be taken into a sprint for development. The working process contains formal activities like dailies, sprint review, sprint planning, backlog grooming and a retrospective which is conducted once a month. The sprints are two weeks long. The sprint review is more like an internal demo where teams present what they have developed during the sprint.

The formal meetings the designer has with the developer are in sprint planning and grooming sessions. The different teams have dailies together but it is mostly the developers talking about what they are doing. Once the designer has completed the ui-designs for a certain task she hands them over to the developer. After the developer has implemented the task they review the result together with the designer

and make changes if needed. The teams have sprint reviews after each sprint but like the daily they are also mostly events where the developers present their ready implementations of their tasks.

Designer 2 (DES2) is working in the team developing the web version of the service. The project itself is still ongoing. Developer 2 (DEV2) was working in the team developing the web version as a lead developers for a year but has recently moved to the role of a cloud architect. DEV2 has a 10 years experience in the software industry and is now a senior consultant with a focus on cloud architecture. He told about his current project where he has worked for a bit over a year as the lead web developer and cloud architect. Designer 2 DES2 has worked in the software industry for about seven months as a UI- and UX-designer. P3 is her first project in the software industry. She has worked in the project for about seven months.

Table 2: Summary of reported projects

	Subject	Project type	Team composition	Project duration
Project 1	DEV1	Invoicing application. Premium product distributed for free. Team develops features on a focus on converting more users into premium users.	4-5 developers, 1 designer, 1 product owner.	Project has been ongoing for 5 years. Developer 1 has been part of the project for 1 year.
Project 2	DES1	A dashboard for a waste management system. The dasboard contains a lot of visual elements like graphs and different charts.	1 project lead, 1-2 designers, 2 developers and a product owner.	Continuous project. Designer 1 was a part of the project for 4 months.
Project 3	DES2, DEV2	Digital consumer service with over a million registered users and hundreds of thousands active users.	Multiple teams developing different platforms. DES1 and DEV2 in the team developing web-version. Web-team consists of 1-2 developers, 1 dedicated designer (1-2 designers from parallel teams), 1 product owner and 1 scrum master for all teams.	A bit over one year. Project is still on going.

4 Results

The results section will be structured in the following way: The first section will present the projects working processes, success and enjoyment. They are presented on the level that they were described by the interviewees. The second section will present how the interviewees felt about their working processes. In the third section we present how the interviewees felt about the collaboration between the developers and designers. The fourth and final section presents what the interviewees see as factors for good collaboration from the perspective of their projects they told about and from their working experiences overall.

4.1 Project success

The success of the projects was not measured in any way. We only present how the interviewees perceived the projects success. The interviewees were asked if they thought that their project was successful and also if they enjoyed the project.

Generally, the interviewees reported that they enjoyed their projects. Good team members and team atmosphere added enjoyment for DEV1 (P1). Other than that the collaboration between the developers and designers was not specifically mentioned in a positive or negative way in the context of the projects enjoyment.

The designers saw their projects as interesting and had positive thoughts about their customers and their customers field of work. DES1 (P2) tough that her project was tough and exhausting specifically because she had to work with complicated designing tasks. She was also overworked and could not allocate sufficient time to every task equally. She also did not feel that she receive enough support from the customers product owner. Overall everyone thought their projects were moderately successful. In P1 The setbacks were seen as caused by forces from outside the team.

DEV1 (P1):

> "I think we've done a good job of converting customers into paying customers, but we also had a lot of setbacks (...) it's not really about us. It's more about our partners who couldn't provide us with solutions that would have to enable us to do something"

DEV2 (P3):

> "I think in the big picture it has been successful. Of course, not every choice has been maybe the best one ever. But in the end, I think that [the service] is progressing well, and we are going to the right direction and there is right people doing the job."

DES1 (P2) thought that because of being overworked by all the ui-design work she had to do alone she could not focus on service design work. This resulted in the customer giving negative feedback about the service design.

DES1 (P2):

> "I realized that because there's lots of UI's to draw, so I didn't have that much time to do the services on par, so it's sucked a bit (...) the feedback was that [the customer] weren't that satisfied with the service design part."

Because of several concurrently working teams working on different platforms there was some unnecessary design work done overall in P3.

DES2 (P3):

> "Yeah, I feel that client is happy. But at the same time, I also feel that we are maybe doing a bit unnecessary work, because the design for the web and design for the apps [happens] at the same time."

4.2 Process flow

In all reported projects the design processes and development processes were mostly parallel with some formal joint actions between the developers and designers. The designers were at the beginning of the processes planning features together with the customers. Once the ui-designs for a feature were ready the developers could start the implementation of it. During the implementation phase developers and designers would collaborate informally when needed until the implementation was ready. This is visualised in Figure 2 where we can see that the design and development work is parallel. This work is done in the same sprints but the design is ahead the development in planning features. This figure is very similar to the processes presented in the research by da Silva et al. [SMMS11] (See figure 3). They present

a high level process based on the findings of their systematic literature review. It is derived from common practices and processes identified from their literature review.

In the framework (Figure 3) the iterations correspond the sprints in Figure 2. The blue part is the development process and the green part is the design process. Figure 3 contains an iteration 0, which was absent in the empirical data of this thesis but present in the literature [SMMS11, SPC14, BMMW15]. Iteration 0 was not directly asked about from the interviewees but they neither mentioned anything similar to it. Figure 3 is similar to figure 2 in that the designers design the ui-designs required for the upcoming development sprint beforehand. The feedback loop and the corrections part are also present in a similar way in the projects presented in this thesis.

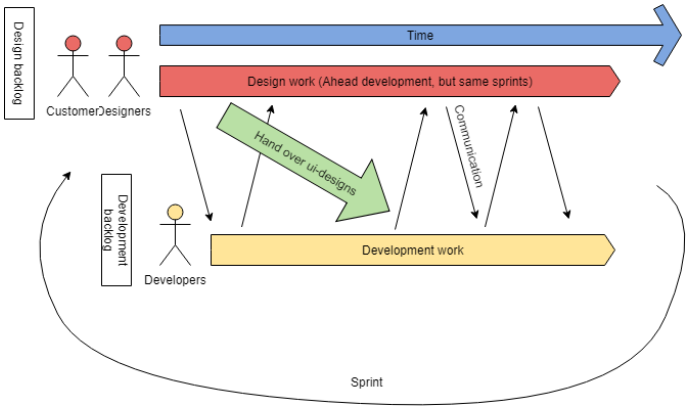

Figure 2: How the process worked on a general level

One issue that DEV1 (P1) and DES2 (P3) brought up in their teams working processes were the high amount of formal meetings. In DEV1:s team they tried to limit themselves of having too many meetings. Every meeting did not necessarily require all developers to take part in them. Only the people who are affiliated with the matter of the meeting are the only ones who should attend, according to DEV1 (P1). He saw that their product owner was experienced and tried to arrange the teams efforts so that the developers could concentrate on development. He did this for example by minimizing the amount of meetings and having the feature specifications and ui-designs well done before the development work started. DES2 (P3) thought that they didn't need any more meetings in their process since they "already had too much".

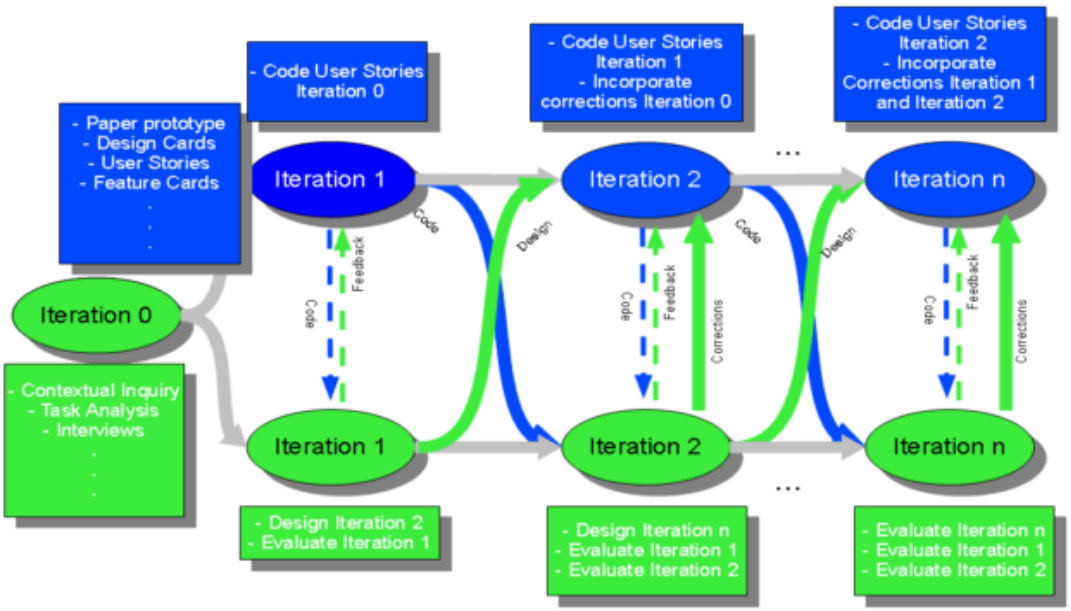

Figure 3: Framework of integrated process between UCD and agile software development by da Silva et al. [SMMS11]

DEV1 (P1):

> "we try to limit ourselves from having too many meetings because it just eats the productivity. We are hired there to produce features, not to sit down in meetings."
>
> "[having too many meetings] happening really easy. (...) in my experience when the product owner is not that experienced, it tries to control anything and everything and when it tries it basically happens through meetings and then you kind of sit on the meetings for the sake of keeping the P.O. up to speed. But we have an experienced P.O. he prefers developers to develop not sit down to meetings."

DES1 (P2) would have liked their process to be more structured, especially if the team would have been any bigger. Since she was the only designer in the team she was able to manage her tasks in a more unstructured way.

DES1 (P2):

> "When we sit next to each other, there's lots of interruptions. So if I would do it again, I would do it so that I would probably try to make slots, so not to interrupt each other all the time (...) So, concentrate those discussions on certain time."
>
> "I wasn't involved with the sprints [ticketing system] (...) When there's many big teams, it's much more important that design is well planned as well. But since we're on the small team, I sort of managed this way"

Dailies, sprint plannings, sprint reviews and sprints themselves were primarily for the developers in all projects. We can see this in the way the processes were structured. Nonetheless, the designers were still involved in some of the activities. DES2 (P3) described dailies and sprint reviews as mostly events where developers talk about their work and present implemented features but simultaneously they were also events where the designers and developers met formally for discussion.

DES2 (P3):

> "We have a daily every day, but that's mostly [developers] telling what they are doing"
>
> "[sprint review] is also mostly developers presenting stuff."
>
> "I'm mostly doing the thinking with other designers and then we have things like sprint planning, grooming for the next sprint. So I think that's mostly the meetings where Iâm with the developers."

Only in P1 were sprint plannings a designated place and time where designers discussed the ui-designs with the developers. After possible adjustments to the ui-designs they then could be handed over for development in the next sprint.

DEV1 (P1):

> "the P.O. and the designer comes to a spring meeting (...) And then they present us with [ui-designs and specifications]. And this is a place for commenting on the design. So first we basically decide if [the feature] cannot work because of these technical difficulties or we give our first initial input and then the designer makes some adjustments. Then when it's approved into the next [sprint], we take the ui-specs and all the other

> aspects that are related to the task and then it's assigned to a certain developer or developers and they start working on it"

In P3 and P2 the designers handed over, presented and went through the ui-designs with the developers whenever the developers were ready to start the development of a feature. This could be at any point of a sprint. The prerequisite was that the ui-designs had to be ready before the sprint started.

DES1 (P2):

> "So in [P2] (...) [sprint planning] was mostly for developers (...) for me, it wasn't that clear. We didn't make a plan for me. I just took care... I just knew what should be ready and when"

DES2 (P3):

> "We can go through [ui-designs] or check it when we are having the grooming or stuff like that. But normally, they'll let us upload to the [ui-design management tool] and explain how it works."

In all projects during the sprints the designers and developers had mostly ad-hoc meetings facilitated by themselves if they needed to discuss the implementation of ui-designs.

DEV1 (P1):

> "Yeah, well we have ad-hoc meetings. There is a basic baseline, so we have a grooming, and sometimes not all the developers participle in that one and that's where they introduced the epics for the next sprint (...) in the one meeting we canât cover [an epic] very deeply and then we just have an ad-hoc meeting with just the people who are relevant to that meeting (...) so it's more of on requirement basis"

DEV2 (P3):

> "Well, usually, if it's a completely new feature or something, the designers usually are the ones who [organise] some kind of [meetings] with the developers. Outside the process, the discussions just happened."

DES1 (P2):

> "[When asked about ad-hoc discussions] Yeah. Always. Like 'hey, this isn't clear, could you help me out?' Then sometimes I have to either explain or actually design something more if I forgot something important."

DES2 (P3):

> "I'm sitting almost next to the front end developer. So it's really easy to go and just ask if there's something that needs to be asked"

DEV2 (P3) saw that it was important that the designs were ready before the development of a feature could start. This was also the way the processes worked in all projects. DEV2 (P3) stated that it would have been a wasted effort to try to implement a feature before it was properly designed. Since the ui-design can change drastically which would mean that the already done programming work would have been redundant. From all projects processes we can also see that this is the way they tended to work, the designs were done in advance.

DEV2 (P3):

> "often, developers would have to wait for the designs to complete or they would ... design would be made in parallel with the application. There's the drawback, that if the design would change and the obligations would need to be changed too if the developer was fast enough to complete the feature before the design was finished so that created extra work. So now we are trying to prevent that from doing the project so that designs are made in advance."

The fact that the team members worked in the same physical location in all projects was seen as a good thing for process success. This way the team members could discuss issues straight away.

4.3 Collaboration between the developers and designers

4.3.1 Close collaboration compared to separation between developers and designers

All interviewees but DES2 (P3) had before worked in projects where the designer(s) was not directly a part of the projects team. In these projects the designer(s) worked outside of the team but provided the developers with ui-designs.

According to DEV1 (P1), it is much worse if the designer(s) is not a part of the team. In those cases the ui-designs need to be thoroughly finished before being handed over because there is greater distance between the developers and designers. This makes the communication more difficult by introducing more latency to the communication chain. If for example there would be something missing from the ui-designs it would be much harder and slower for developing team to ask questions and have discussions with the designers about he ui-designs. According to DEV1 (P1) experiences there is usually some intermediary adding one more step in the communication between the two parties. In those cases the designers are often working on other projects too so the priority to answer developers questions might not be that high.

DEV1 (P1):

> "It's usually much worse (...) there is more distance between the designer and developer, it means that the designs need to be more completely finished. So if it missing something, it introduces a lot of latency into the whole communication chain."
>
> "If the designer is not part of the team, I cannot just walk in and [ask some question about ui-designs]. I have to send an email or [an electronic message]. And usually, [the designer] is not part of the team, it means [the designer is] doing hundred different other stuff so [the question] not put in the priority very high. And then the whole work is basically pending that we get the input from the [designer]."

DEV2 (P3) thought that this separation in the collaboration puts more responsibility on the developers since the developers can't discuss the ui-designs properly face-to-face with the designer. This leaves room for developers interpretations which means that the implementation does rarely end up precisely as the designer had intended. DEV2 (P3) adds that the designers can't take into account every edge case in their

ui-designs (e.g. error messages, invalid inputs, hints and animations) which in turn adds pressure on the developers. According to him, these issues can be avoided with having dedicated designers as a part of the team. Also, This way the end result of the software tends to be more thought out.

DEV2 (P3):

> "Well, it's like a double-edged sword in a sense that it puts more responsibility on the developers because they [receive the specs from the outside]. When receiving from the outside, and not being able to discuss it with the designer that that gives a lot of responsibility for developers to interpret their designs and, well, usually, the implementation doesn't end up being precisely what the designer had in mind, and rarely can the designers take into account every variation. How about if some input in the field is invalid or how do you display error messages, or little hints or little animations to make the experience more smoother?"
>
> "I think that the end result is much better when you have the designer in the team. It's more polished, and it can be revised during the development as well."

However, not having designers as a part of the development team can work for certain types of projects. According to DES1 (P2) if the project is for example a simple website with little to no functionalities the implementation of the designs are quite straight forward. This means that the two groups can work quite independently. Immediately when there are functional complexities in the service it is better to have a team with close collaboration between the two groups.

DES1 (P2):

> "It depends. If it's really basic website, then it depends how complex the system is. So because quite often if there's lots of... in the client side, there's lots of back end stuff which determines what can be shown in UI. So in that sense, I need a lot of work with developers. But if it's just basic website where everything is possible, then I don't think it's that important to go with the developer. So I have done some smaller projects where I can do quite independently. Of course, we talk and we start working together. But it's quite obvious that 'Ok, I will show you something when it's almost ready'. Then there's maybe few small

> things [to fix]. But yeah, with basic website, it's not as important to work closely with developers as a actual service."

4.3.2 Communication

Generally, the communication between developers and designers worked well in all projects reported. Team members were reported positively and no ego issues were reported. Overall team atmosphere was good and it was easy for team members to ask for help and to talk with each other.

DEV1 (P1):

> "overall it's a good team, good people, good atmosphere. So yes, I've enjoyed."
>
> "the chemistry is working so there are no egos involved. If I say that 'I modified this design because it doesn't work', they're like 'okay, it doesn't work, let's modify it'. They don't say it works but you just don't understand something."

DEV2 (P3):

> "[The communication worked] well, generally, well, atmosphere is good, and persons are nice, and it's easy to go to talk to people."

DEV1 (P1) and DEV2 (P3) both reported that good tooling supported the communication. In P1 they changed the that was used for handing over the designs to the developers. With their previous tool they had issues with versioning. The team used a process where designs were linked to relevant tasks in their ticketing system. When a new version of a ui-design was published it was reachable for the developers behind a unique link on the relevant ticket. This meant that tickets had to be frequently updated or they were just simply left with links to outdated ui-designs. This could lead to developers implementing old ui-designs. The new tool had better versioning which eliminated this issue.

DEV1 (P1):

> "the Zeplin tool is really good at [handling specifications]. We used to have in-design and there were a lot of problems of versioning because we always got a when he created a new version it was behind a different

> link. So when the UI specs are usually attached to our ticketing system through the tickets that we use to track our work and those go out of date. And we were implementing the UI with old specs. So with Zeplin tool now, it has improved a lot."

In P3 all designers used the same tools and the designs were handed over to developers in the same format every time in a predictable way.

DEV2 (P3):

> "I think the tooling is also supporting [the communication] (...) designers have been able to merge their designs into one tool. It's really convenient that they come in a standardized format for designers, and they are developing a design system and it is called atomic design pattern."

Everyone reported that some minor misunderstandings and disagreements occurred but those were seen as natural. Developers and designers discussed and clarified together the issues that arose. Some ignorance of each others fields of work was reported. DES2 (P3) though that she did not understand some technical matters about the implementation since she lacks the skills of a developer. If for some reason there is less communication between designers and developers developers DES1 (P2) pointed out that this affects the work in a negative way. Developers might forget or simply leave out minor things from the implementations that still are important for good user-experience.

DES2 (P3):

> "Well, I'm not a developer or coder myself. So there's some things I don't understand how they work"

DES1 (P2):

> "I think [misunderstandings] usually happens when there is no talk. (...) I always want to go through it [when handing over ui-designs] together just to make sure that everything is clear (...) there has been few situations where we weren't together. So it become much more difficult to explain. We have to chat a lot, which takes much more time than talking face-to-face."

Team members all being in the same location eased the communication for everyone. When placed physically close to each other it is easier to ask questions and discuss things. DES1 (P2) especially reported this as an important factor since she can better explain her designs and ideas face-to-face. She thought that explaining designs and their functionalities face-to-face saves time compared to designing every transition, screen and detail. Designers might forget something from the ui-designs therefore it is faster for the developers to ask questions straight away since they are in the same physical location.

DES1 (P2):

> "We were sitting in the same room for four days [in a week] in the client premises. So it was really good, we discussed a lot face-to-face (...) all of us could easily ask each other questions when needed. That's the ideal situation when we're sitting in the same room, of course, because I'm not that fan of chatting in [teams messaging service]"
>
> "I'm that kind of designer, I like to go past quickly. So I do quick sketches and drawings, or UI's, and I rather explain them how they work. Instead of making all the screens of each [view in the] whole flow (...) I think it's faster and saves time for me, and (...) there won't be any misunderstandings. (...) there's also always lots of these talks and discussions because I think it's just best way to proceed."

DES2 (P3):

> "I'm sitting almost next to the front end developer. So it's really easy to go and just ask if there's something that needs to be asked."

The roles of the developers and designers were clear in all projects. Designers had nothing to do with the code. The tools used for handing over ui-designs to developers contained some styling code snippets that the developers could copy if they wanted to. Interestingly designers did not comment on the code but developers commented and gave feedback about the UI-designs so giving feedback on each others work was a one way street.

DEV1 (P1):

> "Yeah, it's very clear. So the designers never try to commit [code] and they don't have any rights. They don't touch the actual code anyway.

> They just do the UI. They have code snippets within their UI tools that we can copy-paste if we want, but they never commit anything. So that makes it a very clear separation."

DEV2 (P3):

> "I think, yeah, we are pretty much the equal level, on the other side, the designers don't do code but developers do give feedback of the designs."

4.3.3 Big picture understanding

Developers and designers had for the most part an equal understanding and big picture view about their projects. This means that they were on the same page regarding the status of the software and the project. The projects had been going on for one year to several years which meant that the teams had familiarised themselves quite deeply with the projects. DEV1 (P1) explained that even though the application they were developing was complex the team members had been working on it for years meaning that they had good knowledge about it. DES2 (P3) reported that everyone in their team had a good understanding of the big picture since in their team they only worked on the user interface side of the application. The developer did not for example work on any business logic happening in the back end of the software.

DEV2 (P3) and DES1 (P2) mentioned that they had visualisations that were used to share information about the projects status. DEV2 (P3) reported that they had a road-map visible for everyone. From the roadmap team members could see which features were coming in the future and be better prepared for them. Even though features were presented on a road-map the designers were at the beginning of the working process from a feature idea to implementation. This meant that they got to know the new features first whereas developers learned about them later. DES1 (P2) had a habit of printing not yet implemented designs and placing them on a wall. The idea was to increase awareness for everyone about upcoming features. Also, before the project started concept design had already been conducted so the team was provided with detailed descriptions of features that was to be designed and implemented. This preparation helped the team get an understanding of what they were about to do.

DEV2 (P3):

> "Yeah. Well, I think everyone knows the roadmap (...) Of course, developers look at it from a more technical perspective that what kind of features and integrations are going to be coming, and designers might think that okay, what kind of parties they need to involve into design and who is it for and what kind of why customers are going to be using it and that. But I think we have been all presented with the same roadmaps and maybe the designers know a bit more since they are in the whole development process, they are at the beginning of the process. So I think they get the new things first and, and then the developers learn about them later."

DES1 (P2):

> "[in P2], it was really clear because since we got the concept already from the other companies. So we had lots of of features [ready]. We started to create it in this MVP style (...) So it was really clear for all of us what's the end result of it is gonna be. The [developers] were really involved, and I printed all the future designs on the wall. So everybody were really aware."

4.4 Factors for good collaboration

The interviewees were asked about what factors they see as important for good collaboration. We present their thoughts in this section. Many of the factors the interviewees saw as important could also bee seen in the projects they reported.

According to the interviewees, the soles were very clear in all reported projects. Especially DEV1 (P1) saw clear role separation between developers and designers as an important factor for good collaboration. In all of the reported projects the roles were very clear. DEV1 (P1) has also worked with designers who tried to implement visual features by committing code. The issue with these cases had been that the code is not usually very maintainable since the designers tended to generate the code within the ui designing tool they used. Generated code works but is not as maintainable as code written by developers whom are experts in that affair.

DEV1 (P1):

> "I have been Working with designers who also tried to commit code. So basically, they try to create code that the new spec could be using as

> such within the code. That is almost always a really bad idea because the code that the developer does is in the perspective of maintenance and of writing clear code. That's their specialty of the developer. The [designers] usually tend to create just UI specs with some tool and it just generates some gibberish code that works, but it's not maintainable. So having that clear separation designers design, and coder code is very important for the long lasting project. If you do a small project that has some very short life cycle, maybe that doesn't matter, but for a product that is evolving and it should go on and live for a long period of time."

Enough time allocation for designers can affect the whole project, reported DEV1 (P1). If the designers don't have enough time to produce the ui-designs on schedule the developers might end up sitting on their hands while waiting for the ui-designs to be completed. DES1 (P2) reported being overworked with designing ui:s which undermined the quality of her other work tasks in the project.

DEV1 (P1):

> "Then also is the time allocation, if the designer doesn't have enough time to usually make the required changes, it means that it's holding the developers back. So, the developers cannot really develop because they don't have the specs, so that creates a lot of latency into the whole product delivery pipeline. And then also you need clear communication channels. Either you are in the same offices or you have some [messaging software] or you have some kind of tool you can put some comments, or you can call so that it's easy to reach out. So it's not like once in a weekly meeting you can do it because then you are basically [in bed to do your work] until the next meeting which is in one week time and then you're sitting on your hands."

As we saw before the team being co-located was seen as important for good communication it was also separately mentioned as an important factor overall. If the designers and developers are not in the same location physically the communication requires other clear channels to work well so that it is easy to reach out to people. DEV1 (P1) reported that these channels can be electronic messaging tools, video meetings and some kind of a way to leave comments on the designs. DES1 (P2) on the other hand thought that digital communication methods should be avoided all together. This is because she thought it is a too heavy way of working and is cutting

the efficiency. Face time is not necessarily required for every day but it should be sufficient. According to DEV2 (P3) the team should be gathered at least three days a week. DEV1 (P1) saw that if the designers meet the developers infrequently the working flow for developers can get disturbed since they might stumble on issues in the designs that needs to be solved by the designers and are hindering their work .

DEV2 (P3):

> "Well, nothing can replace the face time. So you don't need to be in the same space every day but often enough. I would say, ideally minimum three days a week would be good amount to see your teammates and be able to discuss with them."

The physical working space for the team should also have qualities that the designers and rest of the team can utilize to visualize stuff. Qualities like enough walls or boards allow the team to utilise things like physical roadmaps, boards for visualising processes and visualised UI flows. Projects presented in this paper used some of these methods to help in information sharing.

DES1 (P2):

> "I think also, space is important in the [working space]. If I have better chance to visualize stuff and make flows and stuff, it helps everyone sharing information easier."
>
> "I printed [ui-designs] on the wall. So we always have the latest designs on the wall. Because there's lots of people visiting our room, so everybody could quickly see"

Both developers gave strong emphasis on the importance of good software tools for sharing ui-designs between designers and developers. The tool should make the ui-designs easily available for the developers in a detailed format. Meaning that the tool displays information like color hex codes and size of elements in pixels. This makes the ui-designs much more beneficial and the working flow more effective because the ui-designs contain most of the information needed for the developers to implement the ui-design. They need to ask the designers less questions and it minimizes the need for interpretations compared to using methods like screen captures of ui-designs or photographs of drawing boards. Both developers also reported that a good tool also has a practical versioning system within it. This means that the developers always see the latest version of a ui-design and they also have the ability to browse

old versions. This helps in avoiding mistakes by preventing mix ups from having a bunch of different versions of files of a ui-design.

DEV1 (P1):

> "[if you don't have] good tooling, if your specs are a lot like some pdf and with different versions (...) after a couple of revisions you don't know which is the latest and you Ping-Pong with the email [with the designer]. I've done that before all these online tools came available and that was really painstaking."

DEV2 (P3):

> "Tooling is also very important. That the designs are easily available for developers, and tools like Zeplin that you can see hex codes for the colors and all that. Some spacing in pixels so that the designs are specific, they're not like picture taken out of drawing board which leaves so many questions open."

Ui-designs are worthless if the developer lacks the skill set to implement them efficiently. This can be an issue according to DEV1 (P1). Going through the features before implementation together with both parties is important. DEV2 (P3) pointed out that it is a good idea to have some formal process for developers and designers to discuss upcoming features. The idea is that the designers can get a green light from developers if the feature is doable or not and if the designs have to change somehow.

DEV1 (P1):

> "So, if the developers don't know how to properly implement UI specs or you don't really understand the CSS specs very well. So they try to use some lay outing that is very [inefficient]. That also creates a problem. If (...) developers aren't good at their work, that also creates a problem."

DEV2 (P3):

> "I think that there has to be some formal process for discussing. When there are new features coming and it's good that there is some process for working on those, which involves developers and designers, so the

> designers can get the green light from developers if it's doable or not. Or developers can tell that okay, this is doable but it's going to be super expensive to make something. So the design might need to change because it would be too much work to achieve a simple thing."

DES2 (P3):

> "Maybe it's my lack of understanding at some points, what can be done technically. So if there is for example, new feature, I'm looking from the design point of view, sometimes I don't understand what are the options or what is wise to do. I may ask some changes that are not [implementable] or are not the wisest one."

Not only is it useful to detect possible issues before the implementation of features but both designers reported that they appreciate it if developers take part in the designing process somehow. This can be by sparring ui-designs and ideas with the designers or just giving feedback and suggestions. Both designers emphasized the fact that user interfaces and user experience are things that most people have some personal opinions about. Therefore developers can give valuable opinions from their point of view. In P2 DES1 felt that being able to spar with a developer helped her out in figuring out how to design the ui:s which focused on displaying different data.

DES2 (P3):

> "I've worked with some other developer and they just did what I asked. We didn't have any discussion about it, and that didn't work so well. So I think [it's good], when also the developer takes part of the design process and gives suggestions, what could be the best way of [designing] because when you're doing user interface, and user experience, designing the user experience part is something that most of the people have some kind of opinions, personal opinion (...) So I also like to hear developer's point of view."

DES1 (P2):

> "Well, I appreciate when developers are willing to think about design as well (...) so that they would use their own brain too. I appreciate that quality in developers. But not all of them are like that. But when

> there is one who is it is really valuable, because they can bring a different perspective, and it makes better quality. Since I'm the one who likes to talk about stuff face-to-face. So I appreciate also that people are willing to talk, not just in chat [electronically], because that would be in a way too heavy way of working."
>
> "So yeah, since I was the only designer (..) especially with this one [developer], he was smart enough to... I guess he was interested in design, so he wanted to discuss about it. So because the other two developers (...) weren't that interested about discussing with me. (...) [in the project there's] graphs and data and stuff, which is really logical and engineering stuff. So I really struggled with that. So I was really grateful that I got some engineering logical thinking (...) which helped me. (...) Because our product owner was really challenging. He didn't understand why I asked stuff from him. He didn't want to be part of design. He just wanted everything to be ready. So I didn't get any support from him either. So that's why I really needed this developer, and then business designer. So three of us worked together."

Both designers appreciates patience from the developers and willingness to tweak details in the ui. The developers and designers rarely implement the features perfectly on the first go. The ui-designs might change a bit or the designers might want to tweak some details of the feature the developer just had implemented. Therefore the developers must not get frustrated when the designers asks for changes to be made in the implementation. DES1 (P2) talked about how it saves time if the developers implements the feature completely according to the ui-designs specifications. According to her experience some developers tend to implement the features approximately to the specifications. This leads to the designer and the developer having to revisit the designs and perfect the implementation bit by bit. She also pointed out that it brings value if the developers are a bit visual and can spot issues in the designs like badly sized fonts.

DES2 (P3):

> "Patience from the developer. Yeah. Because sometimes there's a lot of like, 'can you please move the thing like two pixels that way or two pixels somewhere'. Also the designs are changing, because this is quite new service, and I'm working with the guy who is also doing the research for the apps. So we are trying to sync the designs all the time. So that

> also means that everything's changing many times now when it's not ready yet. (...) So [the developer] has to do some things many times over again. "

DES1 (P2):

> "I appreciate when or if the developer is willing to tweak the details. So as a visual designer, I appreciate good animations and good details. Some developers might get pissed about putting too much time on details, so I appreciate if they are patient and seeing the value of good detailed design. (...) I once worked with developer who was really precise about the pixels. So quite often developers do something first, it looks visually horrible, and then little by little, it becomes better. But once I had a developer who really followed my guidance and it was almost perfect immediately. So I was like, 'wow', I wish everybody was like that. But of course it save lots of time and so"

The bottom line is that both designers felt that for good collaboration the team should have a safe atmosphere so that people are not afraid to ask questions and discuss things that with each other. Bad attitudes makes the collaboration more difficult. DES1 (P2) reported that she has had experiences where she felt that the developers have sees her as an annoying nitpicker making the developers lives difficult. So it is important that both groups are willing to strive for a as good product as possible and are understanding each others point of view in that strive.

DES1 (P2):

> "I think it's important that developer is that kind of person who is willing to take feedback (...) like [if I] see there's lots of things to fix but if the developer has bad attitude or somehow doesn't want to receive feedback. I think it's important as a designer if everybody wants to achieve good quality it's important that I can feel that I can come and tell those things that I want to fix, not to feel afraid or awkward or stressed or feel guilty about it. So I must be so that... actually for both ways. So if a developer feels that okay this this doesn't seem right or I fail to see it, so it's important that the atmosphere is easy, like good atmosphere. Atmosphere so that you're not afraid to do those fixes. And trust and feel safe. Yeah that safeness.

Sometimes I feel that as a designer, that developers think that I'm just an annoying person thinks only those details are important. Like 'youâre just making my life difficult in purpose'. But I would like that developers somehow won't get irritated about it and realize that I'm not doing it on purpose. I just want to have it as good as possible."

5 Discussion

In this section we will discuss the results, answer the research questions (section 1) using the data from the interviews and compare the results from the interviews to the results from the literature review (section 2). We will also discuss the generalisability and validity of the results. First, lets summarise the data.

The interviewees projects working processes varied between each other but on a higher level they had certain similarities (Figure 2). The designing and developing process were parallel to varying degrees. For example, the projects separate backlogs for both developers and designers. Several typical agile practices like dailies, sprint reviews, sprint plannings and retrospectives were used in the processes. Generally, in all processes the designers first agreed on feature specifications together with the customer whereafter the ready ui-designs could be taken into development. The development processes were iterative processes and fairly independent from the designing processes. The ui-designs had to be sufficiently ready before the development would start. In P2, DES1 had the most unstructured working process, the only requirement for her was to finish the ui-designs in time. During the development phase of a feature developers and designers would collaborate closely if the developers stumbled upon obstacles. In the reported projects the designers mainly worked in the beginning in the process (Figure 2).

Generally the interviewees enjoyed their projects and though the projects were moderately successful. Good atmosphere was reported and no negative remarks were specifically made about the collaboration between developers and designers. DES1 (P2) reported being overworked because of difficult designing tasks and not getting enough support from the customers product owner. Additionally, DEV1 (P1) thought that the designers should have enough time allocated for their work otherwise there is a risk of the work stalling for the developers. DES2 (P3) reported that some unnecessary design work was conducted due to multiple teams working in the project.

Some comments about the processes functionality were made. DES2 (P3) and DEV1 (P1) mentioned that too many meetings can be an issue and that the whole team does not necessarily need to attend all the meetings. Over saturation of meetings eat away from the actual work being done. DES1 (P2) thought that a more structured working process might have been beneficial. Designers and developers collaborated mostly through ad-hoc meetings. Dailies, sprint reviews and such were seen as

mostly for the developers, even though the designers attended most of them. DEV1 (P1) stated that the developers insufficient skills can determine how restricted the designers have to be in designing the features. On the other hand DES2 (P3) stated that she lacks understanding about the technical side of the project hence she is not totally aware of potential restrictions. DES1 (P2) thought that it is important that the developers implement the ui-designs with high fidelity which reduces the amount of fixes that need to be made later.

The interviewees reported about the collaboration in the reported projects and also about collaboration based on their experiences working in the field. Everyone except DES2 (P3) (whom had no experience of it) thought that having the designers as a part of the development team is better than having the designer working separate from the team. According to DES1 (P2), separation is not an issue in small and simple projects like small web-sites. Communication between both parties was reported as working well, minor misunderstandings were seen as a natural part of the communication. No social issues were reported and it was easy for everyone to ask for help and discuss things. A safe atmosphere, not being afraid of asking questions and developers not getting annoyed by designers change requests were also factors seen as important for good collaboration. The developers reported that good software tools are important. A good tool should be able to hand over high fidelity ui-designs to developers and have good versioning of ui-designs. The projects reportedly had clear roles between developers and designers and this was seen as a factor for good collaboration by DEV1 (P1). Both groups are experts in their respective fields and interestingly designers would not comment on the code but developers would give comments on the designs. Developers commenting the ui-designs was behaviour that the designers desired. Both designers reported that they like if the developers spar the designs with them providing a valuable point of view.

The projects teams all worked physically in the same locations. This was also seen as an important factor for collaboration since it enables efficient communication and information sharing. Visual aids like roadmaps and ui-designs printed on walls were used to share information. DES1 (P2) stated that the teams working space should support the use of visual aids by for example having enough empty wall space. Generally everyone had a good understanding about the big picture of the project but the designers usually knew first about new features since they were planning them together with the customers. DES1 (P2) reported that the ready concept work that had been done before the team started in the project helped them in kicking off the project.

5.1 Answering th research questions

5.1.1 RQ1

To answer RQ1 "What are the challenges in the collaboration between designers and developers working in agile teams?" we have to examine the ways the teams worked in the reported projects and what the interviewees reported as challenges.

Finding a fitting process can be challenging. The working process for the team should support the collaboration between the two groups without disrupting their normal work too much. As we can see from the results the working processes in the projects were fairly different in the details. The teams had varying amounts of formal actions in their processes, but ultimately most of the collaboration between developers and designers happened through natural ad-hoc meetings. This was enabled by the teams being co-located. DEV2 (P3) summarises it well how there should be some formal processes to support the collaboration.

DEV2 (P2):

> "I think that there has to be some formal process for discussing. When there are new features coming and it's good that there is some process for working on those, which involves developers and designers, so the designers can get the green light from developers if it's doable or not. Or developers can tell that okay, this is doable but it's going to be super expensive to make something. So the design might need to change because it would be too much work to achieve a simple thing."

Teams have different dynamics and they need to find a good way of working for them. DES1 (P2) reported that her working process was very unstructured. She would have wanted some more structure to the process. DES2 (P3) saw that they did not need any more formal meetings since "we have too much already". DEV1 (P1) saw that there is a risk for having too many formal meetings of which most are unnecessary.

Sufficient designer resources is important for a smooth collaboration. As we can see in the quotes from DEV1 (P1) a smooth working flow between designers and developers require that the designer(s) have enough time allocated to their design work. Since his team was working on a high priority project they designer had enough allocated time to work with the team. DES1 (P2) was overworked in the sense that she had to allocate most of her time for drawing ui-designs neglecting

other design tasks.

Overall attitude towards design affects the collaboration. The developers attitudes towards designers and their practice and the overall social atmosphere plays a major role in the perceived success of the collaboration. As we can see from the quotes in section 4.4 this was seen as especially important for DES1 and DES2. Also the fidelity of with the developers follow the ui-designs affects the efficiency, according to DES1 (P2). It can be a matter of developer attitude how they regard the ui-designs.

5.1.2 RQ2

As with RQ1, to answer RQ2 "What practices are useful in the collaboration between designers and developers in agile teams?" we examine the practices used in the projects and what practices the interviewees reported as good ones.

Design up-front was present in all the projects. Instead of designers working on the ui-designs at the same time the developers were developing the feature the ui-designs were designed before the feature was included in a development sprint. This was especially apparent in P1 and P3 where the processes were more structured compared to P2. Nevertheless the design work was ahead of the development in all reported projects. P2 saw a good start in their project thanks to already existing concept work and planned features.

Co-location of the teams was several times mentioned as a reason for good communication and collaboration by the interviewees. All the teams were co-located. This fact also allowed for actions like ad-hoc meetings at any time, physical visualisations visible for the whole team and overall unobstructed communication that added to the collaboration.

Good software tools for communicating the ui-designs to the developers was underlined for good collaboration by the developers. All teams also used some kind of software tool for handing over the ui-designs to the developers.

Developers taking part in the design work was a feature both designers wished the developers had. Both saw value in developers sharing their opinions and ideas about the ui-designs or sparring over the ui-designs together with the designers.

Using physical visual aids were used in P3 and P2. DES1 (P2) reported that printing designs and placing them on a wall as being a useful way to share information. Physical roadmaps were used in P3 to display the course of the project.

Developers and designers can easily check the upcoming features from the roadmap and take that into account when designing ui-designs and developing features.

5.2 Related work

In this section we will compare the key findings from the background work to the findings from the interviews. From the literature (Section 2) we identified certain challenges in the collaboration between the two groups. The data from the interviews had some similarities with these challenges.

Issues in social relations like power struggles goes hand in hand with the interviewees reports of the importance of clear roles and good social atmosphere. In the background work, power struggles were reported as a challenge in the collaboration [CSM06, JHM14] but none were not reported in the teams of the presented projects. On the other hand good atmosphere was emphasized on a general level. A feeling of safeness was something desired by the designers.

A lack of designing resources was an issue present in both the literature and the data of the interviews of this thesis. A lack of resources was directly reported in P2.

Timing and synchronization issues were specifically mentioned by DEV2 (P3) and DEV1 (P1) but their working processes were such that they mitigated these issues.

Communication issues were not explicitly mentioned by the interviewees but they underlined that the communication was good and important. Having the teams co-located was seen as supporting good communication.

We also identified factors or practices that are favourable for the collaboration. Again, the data from the interviews had similarities with the background work but also a few practices that were not present in the background work.

Design up front was present in all the projects in one form or another. The ui-designs were completed before the development of a feature would start.

Good overall big picture within the whole team was reported as present in all projects. Especially in P2 DES1 (P2) used the working environment by placing printed ui-designs on the walls to make them visible for everyone. In P3 the roadmap was also visible for everyone.

Facilitating collaboration through different practices was done in all projects.

The projects had typical agile practices like dailies, planning sessions and reviews. Most of the communication was conducted through ad-hoc discussions which were spontaneous and not adequately facilitated by one specific person like a scrum master. The teams being co-located made it easy to have discussions about anything at any time.

Good social relations were present in all projects, reportedly. The designers emphasized that good social relations are important for their work. The developers did not especially mention that good social relations are important but they reported that their teams had good social relations. Barksdale et al. reported that healthy relationships makes it easier for team members to ask for help and get support [BS12]. Interestingly, the easiness for asking help was also mentioned in several interviews.

Organisational culture and support was not especially brought up by the interviewees. But we can see from the data that in P1 the customers product owner, representing the organisation, and the team were all on the same page about the process, involving the designer, being used. In P2 DES1 would have wanted the customers product owner to be more active in making decisions in the design process. Since DES1 (P2) did not receive the support she wanted she turned to the developers in finding sparring partners.

The interviews brought up a couple of new points that were not apparent in the background work. These are **the importance of proper software tools** for communicating designs to developers and **designers wanting developers to take part in the design process** to increase the value of the design work.

5.3 Validity and limitations

Validity is analysed using the validity aspects (construct validity, internal validity, external validity and reliability) by Runeson & Höst [RH09]. Since we are not examining causal relationships, internal validity is not analysed.

There are several factors that affect the external validity (generalisability) of the results in this thesis. Generally, agile software developing teams are very different from each other in many aspects. There are several variables that are unique for every project in the field. Some of there variables are for example the customer, what kind of software is being developed, team composition, working processes, team coherence and many more.

The interviewees of this thesis share a somewhat similar background. They are all from the same geographical location, they are consultants and a part of the same sphere of professionals located in the Helsinki metropolitan area. They differ in the amount of work experience they have in the field. The difference is greater especially between the developers and designers. All of the project teams consisted of both developers and designers, they had somewhat similar working processes (see figure 1) with typical practices like dailies, retrospectives and sprint plannings. All projects had 1 - 2 designers in the team. The teams had surprisingly similar issues and ways of working. Naturally, the results from these projects can't be generalised for all agile software projects. The results may apply for projects with similar traits like the ones examined in this thesis.

There are certain points that can be seen as threats to the validity of the results. Most notably is the sample size. Only four interviewees with quite similar backgrounds gives only a very small view of what is happening in the field. The interviewees were all persons the writer of this thesis knew from the industry, this reduces the reliability aspect of the validity.

Some issues regarding construct validity exists. DES2 (P3) and DEV2 (P3) that were a part of P3 gave different and varying amounts of information but not conflicting information. Also the interviews varied in length. Shortest interview was about 17 minutes and the longest almost 45 minutes. The interview process could have been tested beforehand and be more refined to get more information extracted. The interviewees were given a chance to comment the usage and interpretations of their quotes.

The questions for the theme interviews (See appendix A) were grouped under either RQ1 or RQ2. The answers to the questions were thereby grouped under the relevant research question.

6 Conclusion

In this thesis we researched the close collaboration between UI designers and developers in agile software development teams. The research was done by interviewing two UI designers and two developers. Also, a literature review was conducted to get an overview of the earlier work.

The literature showed that UI designers and developers collaborating closely together is a desired way to integrate designing with agile development. However, overall there is no consensus on the field about how to integrate the two practices.

The interviews showed that it can be challenging to find a fitting process for the team. The reported projects combined the design process with the development process in different ways but they all were similar in that the design work was ahead of the development (Figure 2). This was similar to the framework by Silva et al. (Figure 3). A lack of sufficient designer resources was present in one reported project (P2) and was reported as a potential challenge by one developer (DEV1). The designers saw that the developers overall attitude towards design work can be challenging since it affects their work.

Design up-front was distinct in the reported projects workflows. All the teams were co-located and this was also reported as enhancing the communication between the two groups. All teams used software tools to communicate the ui-designs to the developers. Especially the developers reported this as a good thing for the communication. The designers saw it as positive if the developers participated in the designing process, providing valuable point of views. Two projects (P2, P3) used physical visual aids extensively to communicate information about the ui-designs and the status of the projects.

The results of the interviews matched the findings from the literature for the most part thereby strengthening the findings from the literature. Distinct findings from the interviews were the importance of good software tools for communicating ui-designs to developers and designers appreciating developers taking part in the design process.

Compared to the literature this thesis took a more detailed look at specifically the collaboration between developers and designers in a specific settings. In the field there are a wide range of different compositions of teams and several ways to integrate designing with agile software development. An interesting question is, if and how the team composition would affect the collaboration? For example, What

if the team consisted of more than 1 - 2 designers, and how would this affect the teams dynamics? The bottom line is that integrating design to agile development needs more research so that ultimately some general guidelines could be established.

References

BD16 Bordin, S. and De Angeli, A., Focal Points for a More User-Centred Agile Development. 2016, pages 3–15.

BLB12 Brown, J. M., Lindgaard, G. and Biddle, R., Joint implicit alignment work of interaction designers and software developers. *Proceedings of the 7th Nordic Conference on Human-Computer Interaction Making Sense Through Design - NordiCHI '12*. ACM Press, 2012, page 693.

BMMW15 Brhel, M., Meth, H., Maedche, A. and Werder, K., Exploring principles of user-centered agile software development: A literature review. 2015.

BS12 Barksdale, J. T. and Scott, D., Software product innovation in agile usability teams: an analytical framework of social capital, network governance, and usability knowledge management. *Int. J. Agil. Extrem. Softw. Dev.*, 1,1(2012), pages 52–77.

CSM06 Chamberlain, S., Sharp, H. and Maiden, N., Towards a Framework for Integrating Agile Development and User-Centred Design. Springer, Berlin, Heidelberg, 2006, pages 143–153.

DSMS18 Da Silva, T. S., Silveira, M. S., Maurer, F. and Silveira, F. F., The evolution of agile UXD. volume 102, oct 2018, pages 1–5.

GL85 Gould, J. D. and Lewis, C., Human Aspects of Computing Designing for Usability: Key Principles and What Designers Think. Technical Report, 1985.

Hok17 Hokkanen, L., *From Minimum Viable to Maximum Lovable: Developing a User Experience Strategy Model for Software Startups*. Tampere University of Technology, 8 2017.

JHM14 Jurca, G., Hellmann, T. D. and Maurer, F., Integrating Agile and User-Centered Design: A Systematic Mapping and Review of Evaluation and Validation Studies of Agile-UX. *2014 Agile Conference*. IEEE, jul 2014, pages 24–32.

Kuu14 Kuusinen, K., Improving UX Work in Scrum Development: A Three-Year Follow-Up Study in a Company. Springer, Berlin, Heidelberg, 2014, pages 259–266.

Kuu15 Kuusinen, K., Task Allocation Between UX Specialists and Developers in Agile Software Development Projects. 2015, pages 27–44.

MCA16 Magues, D. A., Castro, J. W. and Acuna, S. T., Usability in agile development: A systematic mapping study. *2016 XLII Latin American Computing Conference (CLEI)*. IEEE, oct 2016, pages 1–8.

MGHW10 Mistrík, I., Grundy, J., Hoek, A. and Whitehead, J., editors, *Collaborative Software Engineering*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

Nie92 Nielsen, J., The usability engineering life cycle. *Computer*, 25,3(1992), pages 12–22.

RH09 Runeson, P. and Höst, M., Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14,2(2009), pages 153–154.

SASR16 Sfetsos, P., Angelis, L., Stamelos, I. and Raptis, P., Integrating user-centered design practices into agile Web development: A case study. *2016 7th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE, 2016, pages 1–6.

SK10 Sohaib, O. and Khan, K., Integrating usability engineering and agile software development: A literature review. *2010 International Conference On Computer Design and Applications*. IEEE, jun 2010, pages V2–32–V2–38.

SMMS11 Silva Da Silva, T., Martin, A., Maurer, F. and Silveira, M. Technical Report, 2011.

SPC14 Salah, D., Paige, R. F. and Cairns, P., A systematic literature review for agile development processes and user centred design integration. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE '14*. ACM Press, 2014, pages 1–10.

usa Usability 101: Introduction to usability. URL `https://www.nngroup.com/articles/usability-101-introduction-to-usability/`. Accessed: 2018-10-22.

WMGvdH10 Whitehead, J., Mistrík, I., Grundy, J. and van der Hoek, A., Collaborative Software Engineering: Concepts and Techniques. In *Collaborative Software Engineering*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pages 1–30.

A Interview data

Appenxid A contains the structure used for the theme interviews.

Theme interview

Background information

- Name, age
- What is your education?
- How long have you worked in the software industry?
- Have you worked in any other fields than IT?
- How long have worked for your current employer?
- What is your job description?
- How long have you worked in that role?
- Have you worked in projects where the team included both developers and designers?

A project

- Tell me shortly about a project of this kind
- Team size and composition
- Project duration
- What was your role in the project?
- Did you like working in that project?
- Was there something you did not like in that project?
- Do you feel that the project was successful or troubled somehow?

Working in tight collaboration with ui-designers/developers

- Tell me about the collaboration
- How was it compared to projects where the designer was not a part of the team?
- How did the communication work with the developers/designers?
- Were there communication issues or misunderstandings between developers and designers?
- What did you find challenging in the collaboration?
- Did you all have an about equal understanding of the big picture about the software being produced?
- Were the roles of the developers and designers clear?
- What factors do you see as important for good collaboration?
- What factors were drawbacks for the collaboration?

Practices used

- Tell me about the teams working process
- How did the design process work in unison with the development process or did it?
- What kind of practices did the process/processes contain? (meetings, workshops, sessions etc..)

- What do you think about the practices used? Were they useful or unnecessary? (design practices, communication practices, alignment practices, etc..)
- Was the collaboration actively facilitated? (By team leaders, processes, team members)