Usability Engineering in the Wild: How Do Practitioners Integrate Usability Engineering in Software Development?

Nis Bornoe, Jan Stage

Usability Engineering in the Wild: How do Practitioners Integrate Usability Engineering in Software Development?

Nis Bornoe and Jan Stage

Aalborg University
Aalborg, Denmark
{nis, jans}@cs.aau.dk

Abstract. It has been argued that too much research on usability engineering is incoherent with the processes, and settings being the realities for practitioners. In this paper we want to extend the existing knowledge about usability engineering in the wild. Through 12 semi-structured interviews we wanted to get an understanding of how usability is perceived, and practiced in reality. We found that our participants primarily focus on upfront work to support the design, and implementation process. They implement usability engineering through informal evaluations, and by following a set of local de facto standards. We want to extend the existing body of knowledge about usability engineering in practice, to support the development of methods aimed at practitioners.

Keywords: Usability, usability engineering, user experience (UX), software development, agile development, usability practitioners

1 Introduction

Developers have recognized usability as a crucial factor towards developing usable software [6, 11, 19]. Yet reaching this goal has shown not to be straightforward because usability engineering not is an exact discipline [7, 25], is occurring in interplay with interaction design [3, 11, 28] and software development methods [5, 7, 18, 23, 27] and usability can be difficult to describe in words [7, 10, 11]. Further, the innovation of technology and possibilities of user interaction has dramatically increased, classic post-design evaluation focused and measured upon efficiency and ease of task performance is too limited in today's software development [3]. Recently more qualities have been added to user-centered design. Usability is generally associated with *learnability*, *efficiency*, *memorability*, *errors* and *satisfaction* [22] but notable the introduction of *User experience* (UX) has extended the instrumental values of usability to also include hedonic qualities [2].

When it comes to conducting usability engineering in practice, it has been argued that too much research on usability engineering is incoherent with the processes, and settings being the realities for practitioners [14, 29]. Method validation and studies are often taking place under controlled settings [12] leaving out the factor, that realities

are very varied, and development processes are taking place under very different constraints [14, 21]. Studies about real settings are mainly surveys and case studies, it's suggested that more research takes place in an industrial setting [6] to understand the working context of usability engineering [10], and focusing research on concepts that can be implemented into a practical setting [15, 24].

Our overall research question is: *"How do practitioners perceive and integrate usability engineering in software development?"*

Through semi-structured interviews with 12 practitioners we have built on existing research by investigating practical usability engineering. In this paper we use *practitioner* as an umbrella term for both usability/UX designers and developers.

In the remaining part of the paper we will: summarize related literature (Section 2), provide an overview of our research method (Section 3), present our findings (Section 4), discuss our findings in relation to the related literature and suggest implications for supporting practitioners (Section 5) and finally make a conclusion (Section 6).

2 Literature Review

In this section we will summarize related literature about: 1) *usability engineering in practice*, 2) *obstacles for deploying practical usability engineering*, 3) *the interplay between usability engineering and agile software development* and 4) *new design complexity and challenges*.

A survey showed that software development organizations find usability to be an important part of software development, yet a gap exists between intentions and realities of conducting usability engineering. The importance of usability evaluation is perceived to be less of usability requirements [6]. A study found practitioners on one side expressing value in user-centered design, yet they found difficulties explicating what good usability was, some saw it as a feeling not easily documentable [7].

Several studies have investigated how usability engineering is conducted in practice. Software development organizations do consider usability an essential requirement, but in reality usability engineering is rarely conducted thoroughly [6]. Usability practitioners rarely follow a systematically approach when conducting evaluations [17, 24], for example, they conduct informal user studies [13, 17], and usability evaluations are analyzed informal [24] and according to lightweight and home-brewed approaches [9]. As a result the quality of usability evaluations is often questionable [20, 21]. Lárusdóttir *et al*. report that informal evaluations mainly are conducted to understand context of use and user requirements, and to support design [17]. Furniss and Blandford found that implicit usability expertise is developed through years of practice. This results in *'pre-done' thinking* that can be reused from project to project. Sometimes this experience was saved and shared throughout the organization. They also report that in general lightweight documentation and communication was appreciated because of a fast-paced setting [10]. Regarding methodology, qualitative approaches are more used than quantitative [17, 21, 28] and time constraints is an influential factor when choosing a method [21].

Significant obstacles for deploying usability evaluations in software development organizations have been found to be resource requirements, recruiting participants

[19, 21], missing knowledge and competences, problems with developers neglecting usability perspectives [1, 19], establishing credibility in usability engineering [8, 10, 11], organizational factors [11, 16], for example, often it's not clear who influence the impact of usability [7], and terminology in practical usability is somewhat fuzzy [10, 11]. Documenting user perspectives and usability goals have been found difficult because there are no quality requirements to compare against [7]. Solving identified usability problems is also a challenge [11, 29], for example, it can be complicated to provide design suggestions in complex domains [4, 8]. As development projects are different and include specific constraints and limitations it's not possible to design general methods that apply to all projects [10, 12, 14].

Several studies have investigated the interplay between new development process paradigms based on agile development, and different forms of user-centered design, usability engineering and UX work. The widespread of agility in software development is changing the demands, and dynamics in how usability engineering is conducted [23], as software development takes place in a *fast-paced* environment [9, 27]. Practical problems of effective interplay between design and development has been pointed out by several studies [5, 7, 11, 14, 26, 27]. The challenge is mainly in the difference on how agile approaches, and designing considers, and deals with development projects. Agile approaches are incremental and design needs to keep a holistic view of the entire application [5, 17, 18]. For example, the method *Scrum* is geared towards completing functionality leaving out UX [5] and usability work [7]. A case study showed that conducting usability evaluations in this kind of environment was challenging because only chunks of the design would be completed at a time, making it impossible to evaluate the workflow of a design [27]. Cajander and colleagues found that the limited documentation in agile development, also applied to usability engineering. This made it difficult to document the overall vision of a system and design. Another challenge is measuring quality of usability when no usability quality requirements are stated. As a result all of the above can lead to vague consideration of users, and user involvement [7].

Designing has become more complex, as it's encompassing broader topics on top of making software useable in a specified context of use. Qualities are not only based on task efficiency but also on symbolic and aesthetics [2]. Recent studies indicate that usability, UX, and other design approaches, are used and blended together [16, 26]. Two case studies describe how usability was not considered a standalone discipline. Rather they both considered usability, UX and user-centered design loosely under the same umbrella with no strict lines in between [5, 27]. Designing for complex domains [8] and a wide variety of users in all sorts of contexts, have introduced a set of new challenges. For example, studying in situ is not straightforward since, especially non-work related activities tend to be a lot less defined and happens in many contexts [3].

3 Method

We here describe the: *study design*, *procedure*, *participants*, and *data analysis*.

Study design: we conducted 12 semi-structured interviews with representatives from 12 different Danish software development organizations. Semi-structured

interviews were chosen because this form offers flexibility while still maintaining a structure that makes it possible to systematically compare the interviews, and cover the essential topics. We only recruited participants employed by individual organizations, and did not include participants working as consultants. This was to keep a focus on development within the organization, as part of a team, and not on consulting work. During the initial phase we conducted a pilot study that served several purposes: 1) to test and adjust the interview guide, 2) to better be able to decide which specific profiles would be relevant, and 3) to locate topics of interest. Based on the pilot study changes were made to the interview guide. Two participants did not meet our final selection criteria and have not been included in this study.

Participants: Table 1 provides an overview of our participants. Table 2 provides an overview of the organizations. The participants will be referred to as P(1-12).

Table 1. Overview of the participants. AP means academy professional.

Participant	Role	Education	Industry experience
P1	Designer	M.Sc., IT	12 years
P2	Developer/Designer	Autodidact	16 years
P3	Developer	B.Sc., CS	8 years
P4	Designer	M.Sc., IT	4 years
P5	Developer/Project manager	AP, CS	11 years
P6	Designer/Project manager	Autodidact	8 years
P7	Designer	M.Sc., IT	2 years
P8	Designer	PhD, HCI	3 years
P9	Project manager	M.A, Media	8 years
P10	Chief product officer	PhD, CS	8 years
P11	Chief product officer	M.Sc., Eng.	26 years
P12	Chief product officer	AP, Design	14 years

Table 2. Overview of the organizations. P1-12 corresponds to O1-12.

Org.	Empl.	Primary expertise	Dev. Method
O1	~640	Human resource SW	Informal Scrum.
O2	10	Web-based SW, public, private sector	Informal agile dev.
O3	~60	Web-based SW, public sector	Scrum
O4	4	Web-based SW and apps	Informal agile dev.
O5	10	eCommerce module SW	Informal Scrum
O6	10	One specific web-based mobile solution	Informal Scrum
O7	~120	SW to the healthcare sector	Informal Scrum
O8	~240	SW for mobile devices	Informal Scrum
O9	~50	Web-based eCommerce SW	Informal Scrum
O10	~35	SW to the healthcare sector	Customized Scrum
O11	10	Web-based SW, public sector	Informal Scrum
O12	15	Web-based eCommerce SW	Informal Scrum, Waterfall model

Procedure: all interviews were conducted in Danish and lasted about 40 minutes each. Each interview was divided into two main categories. The first category was related to the type of products being developed and how the development process was

organized. The second category was about usability engineering. Here the questions were centered on how usability was defined, and what purpose usability played during the software development phase. Based on recent experiences the participants were asked how usability engineering was conducted in practice. By having the participants talk about recent experiences, we hoped to get concordant stories, and a broader understanding of the development process, that could reveal more about how usability in reality is being conducted and perceived.

Data analysis: we started out by getting a broad view of the data and locating general and niche topics. This process already started during the data collection, and served, as the foundation for the post data collection analysis. In an iterative process we first went through all interviews, and decided on several labels related to the topics initially found. We then extracted representative quotes from the interviews related to the topics and labels. During an iterative process the authors divided the quotes into groupings. This resulted in a set of groupings that we finally merged resulting in five general categories that reflect the subcategories in section 4. The translation of quotes from Danish to English has resulted in minor changes to the wording and formulation.

4 Conducting Usability Engineering in the Wild

Here we present our findings divided into five different subsections: 4.1) How practitioners understand usability and UX, 4.2) How practitioners set usability objectives, 4.3) How practitioners conduct usability in practice and set local de facto standards, 4.4) How practitioners combine design and development and 4.5) How practitioners report usability problems and act upon these problems.

4.1 Practitioners Understanding of Usability

In the following we present how our participants understood the concept of usability and implemented it into the design process. As recent literature indicates that usability engineering is conducted proactively and closely with other qualities, in particular UX qualities [5, 27], we asked our participants how they would describe usability, about the relation to other design qualities, and how usability engineering was implemented.

Usability is seen as being narrowed down to individual functionality and task completion leaving out a more holistic view.

"For me usability is focusing on the feature without considering the context of use." (P10)

UX was found interesting because it formalizes a set of factors not included in usability, but yet very real when designing.

"It's not just about how easy it is to click a button, it's also about trying to understand the context which the user operates within." (P9)

Usability is not to be replaced by UX. Even that the perceiving and practice of usability engineering has changed over time, usability engineering is still considered important for quality. Rather usability and UX are different and supporting approaches towards the same goal, as a participant noted:

"[Usability and UX] are two sides of the same coin." (P4)

Indicating that usability is reached through a holistic design view, rather than pointing out problems through post design evaluation. We found a clear mutual dependency between usability and UX principles, as the design activities are intervened. P4 noted that that:

"Interaction design facilitates usability." (P4)

From a classic focus on finding usability problems, usability engineering is a mean for reaching and justifying design solutions. Usability factors are to be used proactive just as UX mainly is. A participant explained the following about the role of usability and the interplay with UX:

"For me usability is just one of these areas of competence you must master to create good UX. " (P1)

Proactive approaches towards actively considering usability was a main activity. Rather than focusing on the usability of features, the focus is on making design decisions from scratch that are focused on the flow of a process. When talking about interface design a participant explained the general philosophy when designing:

"Give them an interface that reflects the business process." (P4)

Especially during the initial design phase, the principles of UX seem to support designers, as UX is more geared towards a holistic design. In contrast usability engineering is focusing on more narrow design aspects. By itself usability is seen, as a set of principles used when filling out, and detailing a design frame.

"We conduct workshops where we look at traditional usability values, well, it changes during the project, so early in the project we focus a lot on user experience. When we get further along it's more about getting the usability tuned..." (P10)

This participant further elaborated how usability was used for *design tuning*. When the concept is in place, usability engineering is used to make this design solid.

"We start out with a relative low fidelity. It's all about getting the concept in place. Once we have coded the first prototypes, we hope we have been thoroughly enough during the design phase, so the focus can be on optimization, and less on the concept. Of course we can have missed some essential aspects, but this should only happen on rare occasions." (P10)

Practical challenges hold back the process of actual product evaluation, especially when to evaluate, and what to evaluate. As the objectives and needed outcome is quite different during the different stages of development, it's not possible to conduct the same type of evaluations on mockups, and running prototypes. A participant mentioned that his organizations simply isn't ready for this process:

"The basic problem is that we would want users to test in as realistic a system as possible and that is difficult before it's finished. I do not think we have reached a level where it makes sense to have users testing on mockups of screens or drawings." (P3)

Especially the challenge of evaluating, and testing software during development was pointed out, as something highly unplanned. Rather a common approach was based on trial and error. Design and development would be based on what seemed plausible, and based on experience (more about this approach in section 4.3). The outcome would then be tested in some kind of production form.

"We are trying to come up with something that makes sense based on previous experience from the field, and then we try it out in the wild." (P5)

Ambiguous concepts such as usability and UX were found resource demanding, and tough to thoroughly evaluate during the early design phases. What the designers and developers found reasonable, and how they perceive the usability and UX of a product can turn out to be considerable different from what the users perceive in production settings.

"Sometimes it is necessary that we actually release it into production and get some experience. Especially when it comes to the user experience we might very well say: 'hey, the users perceived the usability and user experience different from what we expected and intended with our design. It was not quite the way the users actually used it.'" (P10)

In summary we learned that usability was considered, as part of the overall design strategy, as opposed to a single process task, and used as a vague term often mixed with other concepts. UX was often mentioned, also this concept would be used in a vague manner. By vague we mean that the objectives and purposes were not clear, and the participants did not rely on a theoretical background, besides the classic five usability goals. Rather it was about a *good feeling* when designing. This made it difficult for the participants to explain what usability and UX meant, and how they were implemented into the process. From the interviews we have learned that the usability engineering methods presented in HCI research have little in common with practice. Yet it's clear that practitioners do informal work based on this research.

4.2 Setting a Foundation for Design

In this section we present how our participants explained usability, and prepared a design. Studies report that setting usability goals was found to be a challenge because practitioners, found it difficult to set requirements that can be compared against [7, 14]. We asked our participants about how they collected domain knowledge, prepared a project and set usability goals.

Especially the benefits of usability engineering can be difficult to explain, and document. For example, providing measurements of the value of conducting a series of usability evaluations, or other kinds of usability engineering, is tough to make explicable, and credible as a participant noted:

"…how does one get usability included into business cases so that they are credible higher up in the system?" (P10)

Instead of focusing on explaining the benefits of usability engineering one approach was to use measurements based on the purpose of the software. For example, a participant specializing in optimizing eCommerce software could show results of increased sales with optimized websites.

Another approach was to give a simple tangible explanation of the initial steps taken to reach a solution. This made the process understandable, and provided essential information, used as foundation of the software design:

"I usually explain the steps of the UX process, I go through during a software development project, to make sure that usability is achieved. In the beginning you have business goal clarification: What is it the business wants to achieve with this? What is their purpose? Whom are they developing to? What problem is it they want to solve?" (P8)

User/customer input was not found to be helpful during initial stages as our participants reported that users need something concrete to relate to.

"We are experiencing that our customers find it incredibly difficult to explain what works. They just don't know until they see it, and it makes sense, so we have difficulty using customer input when designing something new. Customer input is actually only useful when we have the first version ready. Then they can relate to it." (P5)

The customers can explain their business flow and overall vision for IT aided tasks, not the form or concept they are looking for. While providing needed input, part of the design challenge is to sort and analyze user input.

"I do not believe the user is necessarily the best designer. I believe design is a profession that is about taking input, and then come up with something that is more brilliant than you could have thought of yourself, and the user can rarely come up with the best solution." (P10)

Dealing with stakeholders can be an obstacle in the process as they might focus on influencing the development through being part of the process and design phase. The challenge is getting the needed input from the stakeholders, while avoiding digging too much into form and concept.

"You meet with the customer and find out what it is they actually want [...] it will probably be a while before you can turn customers, developers, project managers and all stakeholders in fact, to think the opposite or reverse of what they think now." (P2)

In addition obtaining domain knowledge was a step towards problem domain analysis, as this is the part where it's possible to really get an understanding of challenges, and what to solve.

"…part of designing for usability is to generally understand how people use artifacts [...] Another thing equally important is to have sufficient domain knowledge, so you know what matters, and what doesn't matter for the users in relation to the tasks, they have to solve on a daily basis. It's about investigating where the shoe pinches." (P8)

Domain knowledge was gained through different means such as: phone calls with stakeholders and users, user observations and conversations, learning current IT-systems and processes, and through workshops. This supported several layers in the design process, including information about what problem to solve, the context of use and the users. Domain knowledge makes it easier to exploit user input, as both input sources enlighten from different perspectives. Gaining domain knowledge was a challenge. Firstly, the participants needed access to sources. Practically difficulties such as scheduling with users and discussing the right subjects occurred. Secondly, getting an understanding of a domain can be very time consuming.

Clear measurements for comparing usability goals afterwards was not used, despite some participants seeing this coming in the future:

"We have not actually worked with standard metrics [...] it could well be that we come to a point where we say: 'well it's these metrics', but at the moment we say: 'now we have been through a series of reviews, now we think only trifles are left'. That is what we set as our quality, but we have not systematized at this point." (P10)

Despite having usability goals and using these for setting visions, these visions would not be systematic followed up.

In summary we found that usability not is easily explicable. As a result it's difficult to justify and make creditable. User input and domain knowledge are mutual

dependencies. The participants are highly domain knowledge driven when preparing a design. Domain knowledge was useful during initial stages. User input was helpful for adjustments, feedback and visions. The participants would set usability visions, but none used clear goals or measurements.

4.3 Usability Engineering in Practice

Here we present how our participants conducted usability engineering. Studies have pointed out, that implementing theory into practice is challenging due to the constraints, and limitations of practical settings [5, 7, 14, 26, 27]. This leads to informal evaluations [9, 10, 13, 17, 24]. We asked our participants to explain how they conducted usability engineering, if they involved users, and how they aimed at securing usability goals. We focus on two aspects: *informal and ad-hoc evaluations* and *local de facto standards as usability engineering*.

Informal and Ad-hoc Evaluations

Several studies have pointed out that usability engineering is conducted informal and ad-hoc [6, 9, 13, 17, 24]. This approach is also confirmed by this study. This study also shows, that a trend was to use formative approaches. Summative approaches were rarely used, and when used, they were used as secondary evaluation approaches.

During development the teams need ongoing feedback, especially about specific features. Feedback would be gathered instant, for example, by asking a colleague to try out a feature, or get feedback on a certain design idea. A P2 explained how instant feedback was gained:

"...we have been running around with our cell phones and shown our colleagues the best solution we have found. We give people the cell phone, and a simple task, for example: 'find the menu...'" (P2)

This would generate instant informal feedback that could be acted upon immediately. P2 further explained how the team could setup quickly arranged evaluation sessions:

"…sometimes we conducted an evaluation ourselves. We recruited parents and friends, who really not understood anything, and asked them to find a specific thing, or place on the site. We wrote down some points about how they felt about various things, such as menu structure, search queries, structure of articles and stuff." (P2)

Finding users for evaluations is time consuming, especially finding relevant users. On top of that, evaluation sessions have to be arranged, and appropriate evaluation activities needs to be planned. This informal ad-hoc approach was considered a shortcut that, despite not being ideal, still could help the developers setting a course. P2 considered his own activities during development as part of an ongoing evaluation:

"…I sit and test things. At least the first time, I see how they work on different devices. [...] When I cut things into pieces of HTML and JavaScript, I'm also testing too to see, how things work, and often I make changes to how things work." (P2)

Especially when it comes to small projects this was common. With a strict deadline, a small team and limited usability engineering experience and knowledge,

this lightweight made approach made sense because it was very practical, and did not slow down the implementation.

In one case this was taken to the extreme, as the usability goal was simply: if the software tester did not have any complaints, and the users could use the system, the evaluation was successful.

"…if the tester can understand to test the program, and the users can figure out how to use it, then it's good enough. We have never made usability tests similar to what one does in a usability lab…" (P11)

The teams all worked in very practical manners and needed practical explicit feedback they could act upon.

"When we release a feature into the wild, it's based on our own ideas, and understanding. Afterwards we listen to the customers' reaction, and see if they can figure out how to use it." (P5)

In a practical setting this approach was found to be productive. As we will get into later, our participants did not complete formal and detailed analyses of usability factors, and therefor preferred to try out ideas in a trial and error concept.

In summary, when diving into what informal and ad-hoc means, we found that our participants did not follow a clear plan, or organized planed approached. They would need ongoing feedback, especially about specific features and ideas that cheap, easy, and fast could be reported to the developers and implemented.

Local De Facto Standards as Usability Engineering

We found that the individual organizations had built up a body of principles and standards for individual software types. They would use this body more or less for each relevant project. Here local de facto standards were used as a low budget, and easy achievable strategy for reaching usability goals.

Most design and evaluation was conducted somewhat unsystematically, yet the practitioners had made up their own systematic design rules, they would keep on following during development. This type of usability engineering relied on expertise, experience, and gained knowledge about *what works*, by following good practices, and reuse standards. For example, P2 explained how best practices were buildup over time, and used as a set of principles, when designing websites. This included adding certain elements, and placing them in a certain order.

"...unspoken rules regarding how such a website should look like, for example, pressing the logo should bring the user back to the front page. Such de facto standards should not be changed otherwise people get confused. [...] It's like the red button on the TV remote control. You can move things around on the site all you want, but there are some rules that should always be followed. For example, the search box should be up at the top, so it's easy to reach." (P2)

An organization developing web-based software did not explicit work with usability. Usability engineering was implicit in the form of a set of design decisions:

"…we have made some design decisions that permeates an entire system portfolio [...] when you create a new page, it makes no sense, to begin a major usability discussion with yourself, or with each other, because the whole drill is about getting it to look like the other pages." (P11)

As this organization did not conduct usability engineering, the set of design rules was seen as an effective, fast and cheap approach towards usability engineering. This simple formalization was tangible in an organization with low usability expertise.

A participant developing eCommerce solutions explained, that a lot of research has been conducted when it comes to, what converts into a sale. They would base their development on this research, because it made no sense to keep reinventing the wheel. Instead they relied on best practice models. As a secondary activity they experimented with new ideas to keep building a knowledge base about what works.

"We have made an analysis of how a web shop converts the best. Our design is based on a set of best practice models. For example, we know that a certain percentage of users expect the shopping cart to have a certain placement..." (P12)

In summary we found that these local de facto standards were based on accepted principles, and by harvesting the fruit of gained experience. As the companies each had specializations in certain software types, it was easy, fast and useful, to keep following the same principles.

4.4 Reporting and Acting Upon Usability Problems

In this section we present how our participants documented, and dealt with identified usability problems. It has been reported, that it can be complex and difficult finding consensus about, and solving identified usability problems [8, 29]. We asked our participants how they identified, reported, classified, and solved usability problems.

A common approach to judge the severity of a problem, and later finding a solution, was through discussion among colleagues. Who makes the decision can be quite different from organization to organization. Different roles will make the final decision, such as project managers, business development or a salesperson. At other organizations it's up to the developers and designers.

"…during the small tests we sit down and look at what users have written, and if they all agree. For example, if the placement of the search field, or the left menu doesn't make sense, we'll just step in, and change it right away. Perhaps the project manager will send an e-mail to the customer." (P2)

Overall our participants did not engage in systematically classifications, but this participant explained how they systematically would classify problems. What is interesting, is that they have the classification divided into two phases:

"We look into potential consequences of different problems and add different labels to the problems: Is it about speed? Is it about consistency? Is it about outright failure or risk of error? Earlier in the process, we categorize into other labels. Here it's about looking into different needs. For example, is this specific need something we will go for? Is it something significant that will improve the solution? Is it nice to have? Or is it something that we should take into consideration in the future?" (P10)

A participant explained how they changed the form of documentation from usability problem list, to other formats that are stored in the developer backlog:

"…initially [...] we compiled reports with long lists of usability problems. Now we instead write user stories, or bug reports in line with other system requirements. They are added into our back log, and will be prioritized, and dealt with along with all other requirements of the system." (P3)

Problems with no obvious solution were, as a first step, brought up for internal discussion. In addition inspiration was sought both abstract, as a thought process:

"Inspiration comes by letting a whole lot of input run through your mind… (P5),

and more specific by looking into existing technology and solutions:

"I bring examples of the technology: 'Here is an example of technology or code [...] that's how others have solved it. [...] They have done it. This web shop is running like this.' [...] It is easier to communicate when they can relate to something." (P7)

Several developers explained how they often would be stuck in a dilemma, about choosing between imperfect solutions to a problem. Generally *discussion* was the keyword. This was realized in anything from informally discussing with colleagues, or organized by meeting in workshops to discuss, and review suggested solutions.

"…those people assigned to come up with a solution to the problem will present the solution and get some critic. They will then continue with iterations until we believe, we have an acceptable compromise." (P10)

In the end it's about acting, and we found the teams were driven by trial and error.

"Internally we can discuss issues to death, but we're not getting any smarter than we are right now, so it makes more sense getting the feature released as is." (P5)

This requires the teams to make a decision, which often is a compromise, and then come up with a solution, that at least fulfill the feature requirement.

"…now we develop something that works, so at least it doesn't break. We must accept that there are circumstances, under which it doesn't function well." (P9)

In other cases the root of the problem might not relate to the design, but to other factors. For example, the business model, or the customer might want, or expect the users to behave in a certain way. In one example, a customer wanted a system for selling e-tickets for travelling. However, several travel routes that seemed logical from the users perspectives, were not allowed due to the customer's business model, this lead to confusion. Another example is information that could support the user, but due to security or other reasons, wasn't accessible during the user interaction.

"You can not solve this solely through interface design. When the users are stuck, you can only try to help them continuing the process, for example, by providing a more meaningful error message with an explanation." (P8)

In summary we found that reporting of usability problems was done in a wide variety, with the common denominator being lightweight. This happened in the form of simple notes, presentations, workshops and user stories. Instead of documentation, an ongoing communication was the primary force. This fits into the light hierarchies, the fast pace of the projects, and because this approach made it possible to act upon problems, and potential problems during the process. Complex problem solving is done through discussion, and inspiration from how others have solved similar problems. The business or system logic can be a constraint, forcing less desirable solutions.

4.5 The Interplay Between Usability Engineering and Software Development

In this section we present how our participants mixed design including usability engineering with the development process. This interplay has been found not to be straightforward [5, 7, 18, 26, 27]. We asked our participants about the interplay

between usability engineering, UX design, and agile development, and challenges about implementing a design.

All organizations followed some form of agile approach. This highly impacted the design phase, and the usability engineering. Design and development in practice is very much about taking decisions, and not least settling on compromises. Certain elements of a piece of software have to work. Functions have to be able to do what they are supposed to do, even when they might not do it well.

" All projects run into challenges because there are certain features that cannot be finished, suddenly it meets reality..." (P9)

Designing is an ongoing iterative task, and obstacles during the implementation phase is a guarantee. Some examples of problems were: designs that practically could not be implemented as thought by the designer, for example, an input field that wasn't designed to handle the right type of input, an interface not being able to display specific information, users having troubles using the software, and change is legislation forcing the developers to accommodating new laws and regulations.
Here is the iterative nature of agile development not only an advantage to the developers, but also to all involved in the design.

"There are always things along the way you did not consider during the initial stages, and you now have to take these short comings into account as they arise." (P8)

The iterative process makes it possible to quickly implement changes and revisions. P8 furthered elaborated that:

"The strength of the agile process is, that it's possible to quickly respond to changes, and things you learn along the way, and you are not tied to a requirement specification. Therefore, I am engaging in the development process, so I can make on the spot decisions about the things that have not been thought about." (P8)

As has been reported in the past, a key problem is how to include design work with sprints in agile development. This is also a challenge faced by our participants, and is somewhat opposite to the statement above.

"We have had some challenges in getting the interplay between the agile development, and the usability evaluation to run smoothly." (P3)

This is related to keeping the vision/holistic view of the design and the sprints of agile approaches. A problem pointed out when using an agile approach, is that the focus during sprints can become very narrow.

"We have challenges with Scrum [...] by chopping a project into so many little bits, the focus has moved away from what is being developed, such as ensuring the overall usability of the product, towards achieving delivery of the parts committed to in the sprint [...] To some extend Scrum can become focused about deliverables, rather than aiming at developing as perfect a product as possible [...] we had some challenges ensuring that people are dedicated to the feature being developed, even if it is spread over several sprints, while working under the time constraints of a sprint." (P10)

In summary we found that, an agile approach is a double-edged sword. It allows dynamics between the designers and developers, and makes it possible to implement changes, as part of the process, and changes are always needed. On the other side the focus on sprint completion can become the main factor, with the result that the developers lose track of the holistic view of the software. This occurs because projects are divided, or *chopped* into many parts.

5 Discussion

We here discuss four different aspects of the findings.

5.1 The Merging of Interaction Design and Usability Engineering

Interestingly not a single participant mainly included classic post evaluation usability engineering. Usability engineering was always mentioned in association with UX and interaction design. Gulliksen and colleagues note that:

"You can only design your way to usability, hence, the usability professional must design—i.e. generate interaction design solutions in terms of e.g. concepts, structures, contents and navigation." [11]

To our participants interaction design was the leading force, and usability engineering an *add-on* supporting, tuning and justifying the design choices. 11 of the participants were using interaction design as the driving force for the development. One participant did not conduct interaction design, and very limited usability engineering. All were orientated towards problem solving. Here interaction design adds explicit value in the form of designs of functionality that can be implemented. With 'pure' usability engineering a design is needed before it's possible to evaluate, and the outcome of usability engineering towards product development can be limited. A common part of the interaction design process, was evaluating design ideas. This both generated feedback related to the interaction design and usability.

5.2 Perceived Quality of Usability Engineering

Studies have reported that usability engineering is being neglected due to missing perceived quality, and the mindset of developers [1, 11]. We did not find the same skepticism in our study. We believe the organizations have realized that usability engineering is worth including, even that the amount of usability engineering varied a lot. Because of tight deadlines, the practitioners are forced to deliver, as a minimum, something that at least works, even when the solution is not optimal. By combining interaction design and usability engineering, the organizations not only have the hurdle of fixing problems but also proactively receive important input that is helpful during the development process.

A *good feeling*: Studies have mentioned that usability is considered a *good feeling* hard to document [7, 11]. We also found that our participants had difficulties explaining what makes usability engineering, and interaction design successful. We looked into what is meant by a *good feeling*. Based in using well-known principles and standards, mixed with relying heavily on experience and routines reflected in the local de facto standards, the participants had built up a body of knowledge to implicit facilitate interaction design. We believe this body, by Furniss and Blandford described as *'pre-done' thinking* [10], is essentially the *good feeling*. Further, several participants gained new knowledge through blogs, forums and by exchanging experiences at different venues. We also see this knowledge exchange, as part of what forms the *good feeling*.

5.3 Design Goals and Quality Measurements

Usability was a requirement in nearly all projects developed by our participants, yet defining the objectives for usability quality is quite a different story. This was found to be highly challenging, and is clearly a main obstacle for more explicit and systematic usability engineering. Our participants mainly conducted bottom up designs. We believe this is a reason why setting clear usability objectives are not done. The exception being that the local de facto standards to some extent can be classified as usability goals.

Informal and ad hoc evaluations: Several studies have reported informal and ad hoc evaluations as frequently used [6, 9, 13, 17, 24]. Our participants often engaged in informal and ad-hoc evaluations. Besides running into the same practical problems as described in the related literature, they also had no clear plan, of what to evaluate, and why to evaluate. Further, the need for ongoing input for development is also a reason why, especially the small teams, would conduct on the spot evaluations. It's too resource demanding and troublesome setting up formal evaluations, when feedback is needed immediately. To deal with the ongoing flow on minor unexpected problems with the interaction design and usability, these informal studies were used to test, and shape functionality in the works.

5.4 Implications for Supporting Practitioners

Furniss and Blandford argues that to provide value the usability component must be flexible enough to fit into a wide variety of practical constraints and limits [10]. Based on both the related literature and the findings presented in this paper we agree. We here point out three specific implications, we find to have potential for supporting practitioners. The common dominators are flexibility, and options for customization.

Scale for prioritizing different user needs: As P10 mentioned in section 4.4, his organization used the following scale when prioritizing different user needs:

"…is this specific need something we will go for? Is it something significant that will improve the solution? Is it nice to have? Or is it something that we should take into consideration in the future?" (P10)

The principles of this scale reminds a lot, of the classic scale used for rating the severity of usability problems in the form: *minor*, *moderate* and *severe*. We support the idea of such straightforward scales. They are easy to understand, learn, and can be used by teams consisting of people, with a wide variety of backgrounds and roles. It could be interesting to further develop this scale, and evaluate it in a production setting. We speculate that such a scale could be used for structuring, and facilitating initial discussions, about which features to include in a prototype. We also speculate, that similar scales could be used to facilitate decision-making during other phases.

Facilitating workshops: We found that workshops were used during several phases in the projects. This included gathering domain knowledge, presenting designs to customers, evaluating designs, and internally in the organizations to present usability problems, and when discussing potential solutions. A recent study found that workshops are a popular approach [13]. We find that especially workshops were popular because it's a dynamic approach, allowing customizations to many different

purposes. We also found that practitioners had problems using workshop sessions optimal. This was especially true when including users, because the practitioners experienced problems both running the right activities, and directing the users towards providing relevant feedback. Here we see potential for better optimizing an already poplar method. For example, Bruun and Stage showed how focused redesign workshops supported developers to *look forward,* and focus on overcoming current system limitations, when solving usability problems [4]. P5 mentioned that:

"Internally we can discuss issues to death, but we're not getting any smarter than we are right now…" (P5)

Based on the practices presented in this study, along with the example above, we see workshop facilitation as a promising approach for supporting practitioners.

Local de facto standards as local interaction design patterns: Juristo and collogues proposed making patterns similar to design patterns known from software development, to support the design and development team [15]. Furniss and Blandford found that designers would develop implicit expertise through years of practice. This results in *'pre-done' thinking* that can be reused. They further point out that it's assumed that the goal of usability is to improve systems rather than identifying usability problems. Therefor it's concluded that effective communication of experience and knowledge, is essential [10]. As we found that local de facto standards are highly used, and seem to support the practitioners, we see potential in systemizing this. This could be methods for documenting, sharing, and communicating these local de facto standards within the organization. Such activities could be an approach towards building up a corpus of local design patterns. Potentially such standards can also support setting usability goals that can be used during evaluation, as there will be something to measure against.

6 Conclusion

Through 12 semi-structured interviews, we investigated our initial research question:

"How do practitioners perceive and integrate usability engineering in software development?"

We found that practical usability engineering is highly driven by local de facto standards, and most work related to usability engineering is informal and lightweight. This includes: setting goals, including users, evaluating, and reporting usability problems. Usability engineering is mixed together with other concepts such as UX and interaction design.

A large majority of the development projects are completed within weeks to a few months. Especially this factor is central, as such fast-paced environments do not leave much time for traditional usability evaluations. The participants need ongoing feedback that can be acted upon during development. They see usability engineering as part of interaction design, and interaction design needs to support developers developing functionality. Especially the local de facto standards were seen as an efficient approach towards creating solid designs fast.

When it comes to using an agile development framework, we found that this was a doubled-edge sword. An agile development environment allows dynamics between

the designers and developers, and makes it possible to implement changes, as part of the process. On the other side the focus on sprint completion can become the main factor, with the result that the developers focus less on usability perspectives.

An obstacle listed in the literature review was missing expertise. In this study we found, that an example of this missing expertise is how to effectively use different usability engineering methods. For example, conducting workshops with users could be a challenge. The reason being that it could be difficult running the right activities and getting the needed output from such sessions.

This study was limited to Danish software development organizations. They mainly developed standalone and web-based applications intended for desk and laptops, and they all worked within some type of agile development framework. Therefor these findings do not extend to all organizations and projects.

Based on the findings presented in this paper and together with the related literature we see potential for further investigating, how the implications for supporting practitioners can be further developed, and evaluated in a practical setting.

References

1. Bak, J.O., Nguyen, K., Risgaard, P., Stage, J.: Obstacles to usability evaluation in practice: a survey of software development organizations. Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges. pp. 23–32 ACM, New York, NY, USA (2008).
2. Bargas-Avila, J.A., Hornbæk, K.: Old wine in new bottles or novel challenges: a critical analysis of empirical studies of user experience. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2689–2698 ACM, New York, NY, USA (2011).
3. Bødker, S., Sundblad, Y.: Usability and interaction design – new challenges for the Scandinavian tradition. Behav. Inf. Technol. 27, 4, 293–300 (2008).
4. Bruun, A., Stage, J.: Active Collaborative Learning: Supporting Software Developers in Creating Redesign Proposals. Proceedings of the 5th International Conference on Human-Centered Software Engineering. ifip, Paderborn, Germany (2014).
5. Budwig, M., Jeong, S., Kelkar, K.: When user experience met agile: a case study. CHI '09 Extended Abstracts on Human Factors in Computing Systems. pp. 3075–3084 ACM, New York, NY, USA (2009).
6. Bygstad, B., Ghinea, G., Brevik, E.: Software development methods and usability: Perspectives from a survey in the software industry in Norway. Interact. with Comput. . 20 , 3 , 375–385 (2008).
7. Cajander, Å., Larusdottir, M., Gulliksen, J.: Existing but Not Explicit - The User Perspective in Scrum Projects in Practice. In: Kotzé, P., Marsden, G., Lindgaard, G., Wesson, J., and Winckler, M. (eds.) Human-Computer Interaction – INTERACT 2013 SE - 52. pp. 762–779 Springer Berlin Heidelberg (2013).
8. Chilana, P.K., Wobbrock, J.O., Ko, A.J.: Understanding Usability Practices in Complex Domains. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 2337–2346 ACM, New York, NY, USA (2010).
9. Følstad, A., Law, E., Hornbæk, K.: Analysis in practical usability evaluation: a survey study. Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems. pp. 2127–2136 ACM, New York, NY, USA (2012).

10. Furniss, D., Blandford, A., Curzon, P.: Usability Work in Professional Website Design: Insights from Practitioners' Perspectives. In: Law, E.-C., Hvannberg, E., and Cockton, G. (eds.) Maturing Usability SE - 7. pp. 144–167 Springer London (2008).
11. Gulliksen, J., Boivie, I., Göransson, B.: Usability professionals—current practices and future development. Interact. with Comput. . 18 , 4 , 568–600 (2006).
12. Hornbæk, K.: Dogmas in the assessment of usability evaluation methods. Behav. Inf. Technol. 29, 1, 97–111 (2010).
13. Jia, Y., Larusdottir, M., Cajander, Å.: The Usage of Usability Techniques in Scrum Projects. In: Winckler, M., Forbrig, P., and Bernhaupt, R. (eds.) Human-Centered Software Engineering SE - 25. pp. 331–341 Springer Berlin Heidelberg (2012).
14. Jokela, T., Koivumaa, J., Pirkola, J., Salminen, P., Kantola, N.: Methods for Quantitative Usability Requirements: A Case Study on the Development of the User Interface of a Mobile Phone. Pers. Ubiquitous Comput. 10, 6, 345–355 (2006).
15. Juristo, N., Moreno, A.M., Sanchez-Segura, M.-I.: Analysing the impact of usability on software design. J. Syst. Softw. 80, 9, 1506–1516 (2007).
16. Kuusinen, K., Mikkonen, T., Pakarinen, S.: Agile user experience development in a large software organization: good expertise but limited impact. Human-Centered Software Engineering. pp. 94–111 Springer (2012).
17. Lárusdóttir, M., Cajander, Å., Gulliksen, J.: Informal feedback rather than performance measurements – user-centred evaluation in Scrum projects. Behav. Inf. Technol. (2013).
18. Lee, J.C., Scott McCrickard, D.: Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. Agile Conference (AGILE), 2007. pp. 59–71 (2007).
19. Lizano, F., Sandoval, M.M., Bruun, A., Stage, J.: Is Usability Evaluation Important: The Perspective of Novice Software Developers. The 27th International BCS Human Computer Interaction Conference (HCI 2013). (2013).
20. Molich, R.: The Quest for Quality: Usability Testing Assessment. Proceedings of the 11th Danish Human-Computer Interaction Research Symposium. pp. 56–59 (2011).
21. Monahan, K., Lahteenmaki, M., McDonald, S., Cockton, G.: An Investigation into the Use of Field Methods in the Design and Evaluation of Interactive Systems. Proceedings of the 22Nd British HCI Group Annual Conference on People and Computers: Culture, Creativity, Interaction - Volume 1. pp. 99–108 British Computer Society, Swinton, UK, UK (2008).
22. Nielsen, J.: Usability engineering. Morgan Kaufmann (1993).
23. Nielsen, L., Madsen, S.: The Usability Expert's Fear of Agility: An Empirical Study of Global Trends and Emerging Practices. Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design. pp. 261–264 ACM, New York, NY, USA (2012).
24. Nørgaard, M., Hornbæk, K.: What do usability evaluators do in practice?: an explorative study of think-aloud testing. Proceedings of the 6th conference on Designing Interactive systems. pp. 209–218 ACM, New York, NY, USA (2006).
25. Seffah, A., Metzker, E.: The obstacles and myths of usability and software engineering. Commun. ACM. 47, 12, 71–76 (2004).
26. Da Silva, T., Martin, A., Maurer, F., Silveira, M.: User-Centered Design and Agile Methods: A Systematic Review. Agile Conference (AGILE), 2011. pp. 77–86 (2011).
27. Sy, D.: Adapting usability investigations for agile user-centered design. J. usability Stud. 2, 3, 112–132 (2007).
28. Venturi, G., Troost, J., Jokela, T.: People, Organizations, and Processes: An Inquiry into the Adoption of User-Centered Design in Industry. Int. J. Hum. Comput. Interact. 21, 2, 219–238 (2006).
29. Wixon, D.: Evaluating Usability Methods: Why the Current Literature Fails the Practitioner. Interactions. 10, 4, 28–34 (2003).