Communication through Boundary Objects in Distributed Agile Teams

Johan Kaj Blomkvist
Linköping University
581 83 Linköping
johan.blomkvist@liu.se

Johan Persson
Linköping University
581 83 Linköping
johpe974@student.liu.se

Johan Åberg
Linköping University
581 83 Linköping
johan.aberg@liu.se

ABSTRACT
Personal communication between User-Centered Design (UCD) specialists and developers is important for communicating user needs and design solutions in agile development. In distributed projects where opportunities for personal communication are limited, the design documentation is an important surrogate. This study has investigated the perceived effectiveness of boundary objects in a distributed agile team, and their role in communicating target user needs. Six in-depth interviews with UCD specialists showed that the boundary objects rarely communicate underlying needs of the users but rather focus on interaction with the system that is being developed. The used boundary objects also do not work as stand-alone deliverables; they need to be explained and elaborated. Making the boundary objects comprehensive enough to be stand-alone is seen as too time consuming and not worth the effort. For agile projects with distributed teams, this creates hand-over and follow-up problems.

Author Keywords
User-Centered Design; Agile Development; Distributed Projects; Boundary Objects; Communication

ACM Classification Keywords
H.5.2 User interfaces; K.6.3 Software development

INTRODUCTION
Agile development methods are becoming increasingly popular in software engineering as the complexity of systems and their use of context continue to increase. However, it seems that agile development in itself does not provide the necessary insights or opportunities to realize the usability of a system [11]. The need therefore exists to integrate agile development with the perspective of usability and the methods included in User-Centered Design (UCD). This integration can however be problematic as there is no single unified agile development method. Rather, the term agile development refers to a set of principles for the creation of software. Furthermore, UCD is an umbrella term for a perspective on how usability is accomplished and several methods dedicated to this purpose. As possibilities and obstacles for a successful integration can vary due to differences in organization and project structure, it is necessary to examine and adapt the methods after each individual instance of integration.

The two methodologies also have different perspectives on how and where resources should be allocated [9,29], e.g. concerning the roles employed and how to approach the concept of usability (e.g. [8,12,28]). Furthermore, the two methodologies have different views on documentation. The agile methodology values working software and face-to-face interaction. Conversely, the UCD methodology aims to document the user and the use of the system in as much detail as necessary. This creates the question of what UCD techniques to use and how the result of these should be communicated [5].

The present study was performed in co-operation with a Swedish usability consulting company involved in a large project employing an agile development method. The project was developing an e-healthcare system with the agile method Scrum. The project consisted of several services that, in their own right, could be considered as individual projects. The complete project was on a national level with several organizations and consulting firms involved. This meant that the members of the agile teams were distributed on several locations across Sweden.

The aim of the study was to examine the difficulties, obstacles, and possible solutions in relation to a successful integration of UCD and agile development within the scope of the ongoing project. The focus was on the appropriateness of the used artifacts and deliverables for communicating the results of the UCD work with the software developers in the project, as perceived by the UCD specialists. Based on this, it should be possible to suggest improvements to the communication quality of design documentation. In the study, the artifacts and deliverables are viewed through the concept of *boundary objects*.

Research Questions
In a distributed agile software development project:

1. How do UCD specialists experience the communication with developers through boundary objects?

2. How is the UCD specialists' understanding of the target users communicated with developers through boundary objects?

BACKGROUND

User-Centered Design

UCD is a concept that covers both the methods and the underlying philosophy to create usable systems. UCD strives to create systems with high usability by focusing on the end-user. The term user-centered design is sometimes referred to as human-centered design to emphasize that there may be other stakeholders than users [13]. It has been defined as an *"approach to systems design and development that aims to make interactive systems more usable by focusing on the use of the system and applying human factors/ergonomics and usability knowledge and techniques"* [13]. Design methods such as Contextual design [2], Usability engineering [20] and Goal-directed design [6] have been developed for the purpose of achieving usability and the more general concept of user experience. A common trait for these methods is that they all contain three overarching phases; research (requirements gathering and analysis), iterative design, and evaluation. Representatives for the end-users of the product are involved to some extent in all these phases to secure the product's usability.

Agile Software Development

Most of the agile methods saw the light during the 1990s but were then called "lightweight" methods. These methods originate from early project methods in the 50s and 70s such as Iterative and Incremental design (IID) and Evolutionary project management (Evo) respectively [27]. There is no unified agile method but the different methods contain several common principles, such as iterative and incremental development, adaptive and flexible planning, and a focus on communication and people [15].

Scrum is an agile development method attempting to cope with the complexity of developing software [25] and does this by focusing on a variation of project management values [16]. By placing focus on project management, Scrum is considered suitable to be implemented with other, more programming-oriented agile methods (e.g. Extreme Programming) that provide more detailed work practices such as coding standards. Scrum uses an empirical process control instead of a defined process control to cope with the complexity of software development. The empirical process control consists of three cornerstones; 1) Visibility, 2) Inspection, and 3) Adaptation. Visibility means that the aspects of the process that affects the outcome must be visible for those who control the process. Inspection means that the various aspects of the process must be inspected so that unwanted variation can be discovered. Adaptation means that if the inspection results in finding unacceptable variations that can influence the product in a negative way, the process must be adapted to solve the variations. A basic element in Scrum is the iterative and incremental process which is called a *sprint*. A sprint normally lasts between 7-30 days. Every day during a sprint an inspection is held in the form of a team meeting. At the end of every sprint a functional increment of the system is delivered to the stakeholders who inspect the functionality and adapt the project as needed [25].

Developing software in distributed teams has recently become common place. Such distributed teams could belong to the same organization or make up a more complex configuration of different organizations. Similarly, the actual geographical distance could be in terms of meters (e.g. with teams in different office buildings in the same location), or on different continents with radically different time zones. Hence, the challenges that come with distributed teams vary from case to case. There is, however, a common denominator, and that is the lack of face-to-face communication. This makes coordination and communication more difficult, which in turn can affect project control and product quality [14]. In response to these challenges, a number of practices for distributed teams employing agile methods have been proposed, e.g. using shared and editable wiki pages to communicate progress, separate backlogs for regional customizations, as well as using IM to make communication easier [17].

Integration of UCD and Agile Methods

A recent systematic literature review uncovered five main challenges to the integration of UCD and agile methods [24]:

1. There is a lack of time for upfront design activities. As a result of the inherent flexibility and willingness to embrace change, agile methods oppose that extensive research and analysis precede development. This is in direct opposition to how UCD strives to create an understanding of the user by gathering requirements through upfront user research.

2. Modularizing design activities is difficult. Due to the incremental nature of agile methods and the preference for short sprints with working deliverables, design activities must be broken down into smaller modules to fit in. This may lead to a lack of holistic thinking which may affect the product's overall cohesiveness.

3. Managing the work dynamics between designers and developers. The agile development process requires continual active participation from designers. Hence, good communication between designers and developers is a key to a successful integration [18]. One limiting factor is that some developers are not interested in design and fail to see the value of design activities [1], which may e.g. lead to a reluctance to implement a design as specified [23].

4. Agile development poses challenges for usability testing. For example, scheduling test sessions is tricky since the rapid pace and frequent changes can make early test results invalid.

Table 1: Description of the interviewed participants

Participant	*Work experience*	*Time in the project*	*Typical work in the project*	*Additional information*
P1	5 years	June 2012 – October 2013	Evaluation and prototyping a sub-service. Interviews, personas, scenarios, sketches, and a workshop. Also planning, contracts and procurement.	Worked 2-3 days with the scrum team each week.
P2	6 years	2009 – September 2012	User research, prototypes, evaluations.	Occasional visits and telephone meetings with developers.
P3	6 years	From August 2013	Mainly lo-fi sketches and design specifications.	Has worked as both front-end developer and interaction designer.
P4	2 years	From April 2013	Interaction sketches, design specifications, evaluations and interviews.	Worked in the same place as the developers.
P5	6 years	From September 2013	Interaction sketches, user stories, business requirements.	Occasional visits to developers.
P6	5 years	From April 2013	Target group analysis, personas, scenarios, lo-fi prototypes, design specifications, user stories.	

Frequent testing throughout the development may be a way to overcome the difficulties, but is a costly solution.

5. The agile principle of minimal documentation leads to confusion when it comes to UCD deliverables. It has been pointed out that documenting design rationale is important for justifying earlier design decisions. Also, being able to trace the source of a requirement is important since it can affect changes made later on in the development.

Boundary Objects

The term boundary object was first coined by Star & Griesemer [26] and is used to describe a variety of objects that inhabit different worlds of knowledge. Boundary objects serve as a medium for actors from different cultural, social and scientific backgrounds. A boundary object is "*an analytic concept of [those] scientific objects which both inhabit several intersecting social worlds…*" [26, p. 393]. Furthermore, boundary objects are described as both flexible enough to adapt to individual actors and the constraints of several actors employing them. But they are also rigid enough to maintain a common identity across actors and backgrounds [26].

When it comes to communication and grounding of information in distributed agile teams, there are great difficulties due to the lack of grounding factors available to the teams [19]. Coordination of information depends on knowing what type of knowledge it is necessary to share [7]. It is thus vital to have awareness of team and task activities. This means that the importance of certain boundary objects, through which the team members can communicate, grows as they should also include a mediation of the underlying activities for the boundary objects. Furthermore these boundary objects become important as a reference for other communication in distributed agile teams as they allow actors from different communities to negotiate meaning and create a shared understanding [19].

METHOD

This study took the form of a case study, where the phenomenon under investigation is the communication through boundary objects as a part of the larger context of integration between UCD and agile methods. The context in which this phenomenon is being examined is the project where the consulting firm's UCD specialists were involved.

Project and participants

The usability consultancy got involved in the project in 2009, with the development of a platform for sharing digital health information. The project later transmuted into a larger project consisting of three sub-services in 2012. In each project, the developers followed a variant of scrum, using two week sprints which the UCD specialists were not part of regularly. The interviews were conducted in April 2014 with seven UCD specialists from the consultancy. One participant had not been communicating with the software developers and was excluded from the study. Four participants were currently working in the project, and all had been working as interaction designers at some point of the project. Development was carried out in different Swedish cities within the same time zone.

Procedure

The interviews were semi-structured and aimed at investigating the UCD specialists' experience of communication with developers through boundary objects. Four of the interviews were conducted face to face and two were conducted over a video call with the program Skype. The interviews were conducted in Swedish and audio was recorded for all the interviews. A template of questions was produced with the relevant previous research in mind to guide the interviews.

- What documents and deliverables do you produce for the developers?
- For every document/deliverable mentioned,
  - What is the communication purpose?
  - What is the content?
  - Is the communication effective?
- What pro's and con's do you experience with communicating through the mentioned documents and deliverables?
- If and when do you use the documents/deliverables as reference for other types of communication?
- If and how do you document your design decisions?
- How do you convey your understanding of the user in the documents/deliverables?

Each interview began with informing the participants and acquiring informed consent to participate in the study. The interviews were later transcribed in Swedish and subjected to content analysis [10] (excerpts have been translated from Swedish to English). Meaning units in the transcribed interview material were identified based on the research questions. For each question, the units were then condensed and interpreted. The analysis proceeded by identifying the common themes in the different interviews and comparing the answers. This step led to a further abstraction into emerging themes, which are presented in the result below.

RESULTS

How do UCD specialists experience the communication with developers through boundary objects?

The boundary objects discussed in this study were all dependent on verbal communication. For distributed agile projects, the limited communicative function of the boundary objects creates problems with both hand-over and follow-up. If the hand-overs are made without personal communication and the possibility for a discussion, much of the intent and solutions risk being overlooked or misunderstood. If the hand-over is final, i.e. there is no more time scheduled in the project for designers, then a lot of knowledge about the user and associated motivations for design suggestions will be unavailable to the team.

The results for this research question have been categorized according to seven different themes. The themes contain answers to what documents and deliverables are used and how, what their perceived purpose is, and what their experienced advantages and drawbacks are.

Potential Boundary Objects

To uncover the participants' experience with boundary objects they were asked what documents and deliverables they produced. The results were quite similar, with some differences due to variations of when they became involved in the design of a sub-service in the project. The different documents and deliverables they produced, and therefore the potential boundary objects, were; (1) Personas, (2) Scenarios, (3) Effect maps, (4) Sketches, (5) Design Specifications, (6) Prototypes, (7) Evaluation summaries, and (8) User stories. In addition to their own produced material (9) demo pages, created by developers also came up as a potential boundary object.

Different and Multiple Purposes

Some of the mentioned documents/deliverables have a single and clear purpose, others are used with multiple and different purposes. A few of the different types are viewed as work artifacts first and foremost and not communication material i.e. boundary objects. Personas and Scenarios were primarily viewed as working material for the initial design. The purpose of the personas was mainly seen as documentation for requirements gathering and the design process by identifying target users, prioritizing these and to convey an understanding of them. Scenarios were also primarily seen as documentation within the design phase and were used to represent possible future use situations based on personas and as a base for extracting interaction flows. Due to the focus on aiding the design process, personas and scenarios were seen as poor communication material to anyone not directly involved in the design process. The structure of the persona was created so that the creator, who also conducted the research, would later easily recall the most important elements.

This is shown by a statement from P1 "*(Personas) are hard to digest, even I experience this if someone else has created it. When I've made a persona I know exactly what parts I've created it from*" and "*personas are hard to extract information from in the text form when they are 2-3 A4 pages long. But at the same time you make them long so that you as a designer can use them later when designing, but then they might not work as well for communication purposes*".

Prototypes were also reported to have multiple purposes. In addition to communication artifacts (which will be discussed later) they were sometimes primarily used as evaluation tools. This reportedly affects the structure of the prototype as details in the prototype are held back to enable more responses from the participants in evaluation. By keeping the detail of the design at a lower level, participants can concentrate on the functions being evaluated and not be distracted by the details. Further it is reported that low fidelity gives the evaluating participants the feeling that the design is not carved in stone. It becomes apparent that as documents and deliverables like personas and prototypes are structured for the prioritized purpose, the suitability for other purposes diminishes.

TAGRI Principle

The TAGRI (They Ain't Gonna Read It) principle was indirectly or directly present in all interviews in relation to personas, scenarios, and effect maps. Though all of the documents and deliverables were submitted to the project database where everyone involved including developers had access, there was some uncertainty of what developers actually saw and used. P1 and P2 expressed the experience that none of the developers read material like personas and scenarios when sent out over email. P1 and P4 further expressed that there was an overflow of material in the project database that was difficult to filter through and that this rendered the database non-functional. Although accessible to developers, P3, P4 and P6 also reports that by their experiences neither of the personas, scenarios or effect maps was used by the developers. P1 and P2 also reported that they didn't see any use of the personas, scenarios or effect maps for the developers.

Due to the experiences that these artifacts and deliverables are not used by developers; personas, scenarios, effect maps, and evaluation summaries are *not* viewed as boundary objects between UCD specialists and developers.

Purpose as Boundary Objects

After excluding personas, scenarios, effect maps and evaluation summaries from analysis as boundary objects the remaining documents and deliverables are sketches, design specifications, prototypes, and user stories. The analysis of the interviews has identified the participants' view of the purposes of each of these. All of the participants reported using sketches as deliverables to the developers. The sketches were reported containing requirements from the scenarios and were used to visualize these. The main purpose of the sketches was to create an understanding of user flows, the interaction with the system and its functionality. It was also seen as a communication aid by grounding and structuring the verbal communication. Lastly its purpose was seen as providing instructions to the developers and to raise changes to the design.

A design specification was used by all participants except one. P5 reported using sketches instead because of arriving at a late stage in the sub-project and that there was no time to produce a design specification in its full extent. A design specification was reported by the other participants to contain sketches coupled with descriptive texts clarifying the interaction and system responses. One of the delimitations of the design specification was that it was not allowed to be interactive as it had to be printable. A design specification was also reported containing graphical design and the sketches used seemed to be hi-fi to convey the detailed design.

As P3 puts it "*We have created a design specification that basically describes every page of the service…with more or less finished windows in relation to design…*"

The purpose of the design specification was reportedly to communicate an understanding of the interaction and the system's response. It was used to eliminate and forgo questions but also to minimize misunderstandings. Further purposes were reported being to create quality and to produce criteria for verifying the implemented code.

P1, P2, P3, and P6 all reported using prototypes as a deliverable to the developers. The prototype was mostly seen as complementary to the design specification as its purpose also was to convey an understanding of the interaction and to minimize misunderstandings. But as an advantage P2 expresses the experience of prototypes as the strongest visualization tool. P3 reported a similar experience with prototypes as conveying a better overview and a stronger feeling for use than the design specification. Furthermore P6 reported that when using hi-fi prototypes they could almost replace the design specification.

P3, P5, and P6 reported using user stories as deliverables to the developers. P3 reported using them as a complement to the design specification and prototype with the purpose of describing possible interactions to each part of the service. P5 used them as complements for the sketches that replaced the design specification with the purpose of providing the developers with finished work packages. P6 reported translating the design specification to user stories with the same purpose of providing finished work packages to the developers.

Neither Comprehensive nor Stand-Alone

When the participants were asked how they experience the communication through documents and deliverables all of them pressed that the boundary objects were not comprehensive and therefore not able to stand alone as communication. As the boundary object's main purpose seem to be to communicate an understanding of interaction, underlying and explicit motivations were excluded. Further and as P6 reports, the process of arriving at a final design is iterative and ongoing throughout development, and the deliverables thus only as comprehensive as the design is final. P3 reports that since the deliverables can contain heavy texts it will always be ambiguous to some extent. As P4 points out, questions concerning the delivery will always arise. When trying to achieve the best possible result, the participants experienced that they always have to verbally communicate the intentions of the document/deliverable as well. P3 therefore experienced the need to deliver in person. P6 expresses a similar need with a face to face review of the deliverables. The notion of the boundary objects being neither comprehensive nor stand-alone is further evident as the different purposes of each boundary object is closely related to being a communication aid.

Time Consuming Boundary Objects

One of the emerging themes was that the boundary objects were experienced as very time consuming by all of the participants. This was reported in relation to the creation of the documentation/deliverables, maintaining them, and reviewing them when delivered. The participants experience that the deliverables could become very large and heavy. This made them difficult and time consuming to package for delivery. While wanting to explain in as much detail as possible to ensure the design, a balance has to be made between the comprehensiveness and the time spent on creating them. P6 describes this with "*it is a challenge to put together a material which is comprehensive enough but not too large*".

In relation to maintenance, P4 and P6 describe their experiences with updating sketches and design specifications as time consuming. Both further expressed that if they were given time initially to create a hi-fi prototype, time could later be saved on maintenance as master components can be used in prototypes. The experiences of the participants suggest that the more complex boundary objects become, the harder they are to interpret and get acquainted with. P6 points out that as a design specification grows the ability to convey an overview of the design as a whole diminishes.

Direct "Translation"

The participants had asked the developers what deliverables they needed and how they structured their tasks from the deliverables they received. The participants thus started to deliver user stories as complements to existing deliverables. This was reportedly done as it would otherwise require developers to have long meetings to turn deliverables like sketches, design specifications and prototypes into user stories as their usual structured tasks. In addition to lengthy meetings P5 also reports the experience with user stories having too much of a technical focus after these meetings and therefore saw the need for the UCD specialists to directly translate their work into user stories. This provided developers with finished work packages and allowed them to focus on the prioritization of the user stories in the product backlog instead of their creation. By creating user stories some of the participants could directly translate design issues, user research like interviews, and evaluations into easily interpretable deliverables for developers. It was also reported that due to time constraints in the project direct translation also occurred with sketches, as is shown by P1's statement: "*You don't put in a week's work on creating a nice presentation of evaluation results because it is more important to get the design going and update sketches*."

Summarizing the results concerning how communication was experienced through boundary objects:

- Personas and scenarios could not be considered boundary objects but rather documentation of user research.
- Prototypes were used for multiple purposes, but mostly to test the design. Prototypes that were interactive could replace Sketches and Design specifications if they were hi-fi enough.
- Prototypes resulted in evaluation summaries, which were not read by developers.
- Effect maps were not read by developers.
- Sketches and Design specifications communicated requirements from user research, such as functionality, interaction, and flows.
- Personas, scenarios, effect maps and evaluation summaries could not be considered boundary objects. However they were important tools for the designers who felt that the value of these techniques lie in the process of creating them.
- User stories were used as boundary objects, sometimes as a way of clarifying Design specifications or Prototypes, or as a way of providing finished work packages

How is the UCD specialists' understanding of the target users communicated with developers through boundary objects?

The main finding concerning communication was that the objects in this study lacked motivations for what they communicate, and why the design looked like it did. This had to be communicated separately, which the participants did when developers requested more information. For instance, while personas and scenarios might be mentioned in verbal communication, it appears that the participants only used the existence of these to keep their backs clear if a design decision was questioned. Three themes emerged.

Problems Stronger than Solutions

The experiences reported by some of the participants were that usability problems seemed to stronger communicate an understanding of the user than suggested solutions would. P1 uses the example that a video recording of the user evaluation could much clearer show the frustration of the user and would therefore convey a much stronger understanding of the user. P1 continues with "*I believe that what would create more empathy is to involve the developers to a larger extent in the requirements gathering where they could meet the user…*" While P2 points out that personas and scenarios contain a lot of information about the understanding of the user, evaluation results were experienced to be the strongest means to communicate the understanding. P3 and P4 also report that interviews and evaluation results were the primary motivation for how user stories were created.

"Translation"

As previously stated, the documentation and deliverables created to convey an understanding of the user is primarily viewed as working material for the design process. This coupled with the time shortage experienced by participants meant that they felt the need to directly translate the understanding of the user to actual boundary objects such as sketches and user stories. P1 and P2 report on the view that it is not the developer's job to understand the user and therefore does not need to care. As described above, information concerning the understanding of the user acquired in interviews and evaluation was directly translated into user stories and sketches. Information about the understanding of the user which could be found in scenarios and personas was also translated into sketches and user stories respectively. In P5's sub-project, user research was missing due to time limitations and existing user stories had a technology focus. P5's response was to get involved with the creation of user stories. This was done to restore what P5 perceived to be the user flows, which would improve the understanding of the user.

On Demand

As it has been shown now, the documentation and deliverables directly containing information for the understanding of the user is not perceived by the participants to be received and read by the developers. Furthermore it has been reported that there are no explicated motivations for design decisions in the actual boundary objects to convey an understanding of the user and thereby the design. Instead the participants' experiences indicate that the understanding is translated into boundary objects as the information is not seen as relevant for the developers to do their job. However, the participants report that they later verbally communicate the understanding of the user if necessary i.e. on demand by developers. This verbal communication seems to be aided by the creation of personas and scenarios as participants report that they sometimes refer back to the existence of this documentation. What emerged in the interviews was that any connections between an understanding of the user (i.e. rationale) and the actual design is internalized in the participants and only externally verbalized when requested. As P2 and P6 states, this is done to cover their backs.

DISCUSSION

There are some possible implications of these results. The risk involved is that some design documentation is not useful. If the intention is that methods are used to communicate e.g. the end user's perspective (persona, scenarios), then those methods must be developed based on a realistic understanding of design work in various contexts, such as distributed development. Otherwise they will not be used, or worse, when they are used they simply take away time from more meaningful activities. When design tools are not used as intended it can lead to: the design is not being implemented in the intended way [25] and the voice of target users being lost during development. Both problems are associated with lower usability of the final product. This means, in some cases, that the role of design methods as boundary objects in development projects should be reconsidered.

It is possible that the reason why the objects are not used or read by developers is a lack of understanding or perceived usefulness of design work among developers. This affects the relationship between developers and designers, which is imperative for the willingness to implement design [23]. The relationship between designers and developers is important [18], especially if design methods are considered as aids that support interpersonal communication rather than stand-alone boundary objects.

There are two approaches possible in relation to the finding that the boundary objects are not stand-alone: 1) accept that boundary objects are communication aids rather than self-explanatory, or 2) try to change or develop new boundary objects for integration of UCD and agile development. We discuss the implications of each approach below.

Seeing boundary objects as communication aids

Accepting that boundary objects are complements to be used as support for coordination, collaboration and communication would mean that time have to be made for those activities in agile projects. This is in line with the sixth agile principle which states that the most efficient and effective way of conveying information to developers is through verbal communication. When verbally communicating their understanding of the users, the participants reported using boundary objects as communication aids but also referring back to the existence of personas and scenarios.

The finding in this case that there were too many documents, and that they were difficult to understand and overview, further supports the approach that the boundary objects should be seen as communication aids rather than stand-alone design documents. But when the agile team becomes distributed, communication and the grounding of information become more difficult. This puts a heavier load on boundary objects as references in other communication.

One obvious improvement for co-located teams is to involve both designers and developers in all activities [5], including user research. P1 expressed the need for developers to be more involved in the requirements gathering to get a better understanding of the user. This is further illustrated in [21] which reports on the use of a workshop with developers and stakeholders to perform user research and requirements analysis. It is claimed that this enabled a distribution of knowledge for the user throughout the team and that the resulting prototype could function as a communication aid later on [21].

This leads to the question of how e.g. personas and scenarios can be made relevant also for the developers. One of the projects in this study involved the whole team in creating user journeys (similar to personas and scenarios) as not doing so in the past had created problems. This however seems like a good idea in theory but that was less successful in practice, since so many projects fail to do so, and it is only feasible if the team is not distributed.

As [4] points out, for an integration to be successful, UCD does not only need to be integrated in the agile methods, agile methods should also be integrated in UCD. This should be done to allow a better integration of the underlying values. Not allowing enough time to adequately present the results of an evaluation could be seen as a failure to integrate the underlying values of UCD in agile development. The lack of a deeper integration on the abstract level also becomes apparent as P1 and P2 claim that it is not the developers' job or responsibility to care about and try to understand the target users. According to the agile principles the highest priority is to satisfy the customer. This should obviously include knowledge of whom the developers are actually building a system for.

Improving the design objects

The other approach is to interpret this research as showing opportunities to develop or tweak design documentation so that it can be more useful in agile development in general, but also more specifically for distributed teams. Such techniques must be both lightweight enough for designers to use them and comprehensive enough to satisfy developers.

Boundary objects in the project

Many potentially useful documents for communicating about users were created and used only by and for the UCD specialists. Naturally, objects that are created and used exclusively by designers do not improve collaboration and communication [5]. An alternative could be to train designers in customizing the objects for developers, rather than for themselves. If working in a distributed agile team, more time has to be spent on making the personas and scenarios easier to interpret. However, while it is important to integrate design work with the agile process [4], not all design work, including tools sketches and other documentation must be shared. Some tools are useful only for designers, just as other aspects of development do not necessarily concern designers.

Personas, scenarios and evaluation results

There seemed to be no physical use of personas and scenarios for aiding the communication with the developers (they are only verbally referred to). Pruitt and Adlin [22] described the persona lifecycle and pointed out that one of the most difficult tasks is the transition from creating a persona to using one in an organization. They stress the fact that personas have to be presented progressively in small and easily interpretable chunks. More importantly, the personas have to be structured after the intended receiver. This means that when creating a persona it must be clear how to access and retrieve information.

Translation also occurred from scenarios and evaluation results to sketches. As deliverables were seen to be neither comprehensive nor stand-alone, sketches could be presumed to be a preferable communication aid. This is in line with [3], where sketches are identified to be well adapted for focusing attention, grounding communication, recording design decisions and to explore relationships within design.

User stories

Some participants started using User stories as part of their communication. This shows the importance of the empirical process control which is a foundation in Scrum. Although not contingent on being co-located, the inspection of what developers needed to structure their work lead to an adaption of the deliverables to include user stories. The use of boundary objects such as user stories and sketches as communication aid is, as previously mentioned, supported by [3]. While sketches are suitable for visualizing and grounding the communication, user stories are well suited for making problems apparent and connecting designs to user groups. As stated in [19], the importance of boundary objects as communication aid grows as teams become distributed.

Sketches and User stories were actually part of a coping strategy that designer used in this study. The limited time for design work during a sprint meant that user research needed to be directly translated into some format that the developers would use. Hence, the participants started creating user stories to complement the other objects. User stories are already part of what the developers use, but focused on technical issues. This was an opportunity for the UCD specialists to make design more prioritized.

The advantages of having designers make User stories are that it complements the technical focus, and that it provides a way to directly translate UCD work into a format that is already used in agile development. However, they seem insufficient to fully communicate an understanding of the user.

According to [7], the understanding of what is critical to share increases as the team becomes co-located. From a boundary object perspective, user stories can be seen as standardized forms [26] as they are structured after user role and goal (e.g. “*as a [user role] I want to be able to [goal]*”). User stories thereby show an advantage of using the boundary object type standardized forms as communication means. User stories, created by UCD specialists, appear to provide the developers with a more direct translation of the UCD work than other deliverables like design specifications and prototypes as they already are an integrated part of the agile development. According to [2] user stories can be seen as relatively detailed definitions of system functions which are derived from user goals. In addition, user stories can connect the design to user categories and user goals [3]. Despite these previously reported benefits, none of the participants expressed that user stories communicated an understanding of the user.

Adding a motivation to the standardized forms might be a good way to provide the missing information about the users' motivation and needs: “*as a [user role] I want to be able to [goal] because [reason]*”.

Prototypes and design specifications

The experience of time consuming boundary objects related both to the creation and maintenance of them. In particular the design specification was reported to be time consuming in its creation because of heavy texts and in maintenance due to the need to update the sketches individually. P6 mentioned that sometimes, and if there was enough time, a hi-fi prototype could be created and function as a replacement for the design specification. What is represented in text in the design specification could possibly also be easier to represent in a hi-fi prototype. While the design specification with its static sketches allows the representation of specific flows within the design, coupling the prototype with user stories could just as easily convey this information.

As both the design specification and prototype was seen as neither comprehensive nor stand-alone, verbal communication was anyway needed to clarify the artifacts. The possible benefits of using a prototype for this purpose are that the prototype is reportedly easier to interpret and gives a clearer overview of the design. Furthermore, it was seen as a stronger communication material because of its interactive qualities. This finding emphasizes the usefulness of prototypes as boundary objects between designers and developers. Presently however, most prototypes also lack motivations for decisions and knowledge about users.

Design rationale

A design rationale as an additional boundary object could possibly be used for the purpose of bridging the design and the personas by externalizing the participants' internal connections. This would, hypothetically, not only relieve the UCD specialists of holding in the connections but also solve the access-related difficulties with the personas. The solution of a design rationale is not necessarily one created for the developers' ease of access (i.e. retrieval). A design rationale in the argumentative perspective could also be used to aid UCD specialists in the design process and therefore possibly help clarify what is essential to motivate. As suggested in [22], the persona should be adapted for the intended receivers. The persona could therefore be re-structured from the design rationale to possibly provide a better aid for communication with developers.

Boundary object repositories

From the interviews it was concluded that all documentation and deliverables were distributed through a project database or a product backlog. The project database and the backlog can through a boundary object perspective be seen as a boundary object type repository [26]. Distributed agile teams puts a heavy weight on the functionality of repositories and the need for easy access in them. The difference between the project database and the backlog seems to be that, while the functionality and easy access of the backlog is apparent as it is deeply integrated in the agile method, the project database failed, leaving some documents and deliverables unseen by developers.

LIMITATIONS AND FUTURE RESEARCH

This work covers one project in one company and the results cannot be generalized to other settings. The findings are limited to the UCD specialists' experiences because project management declined involvement of the developers. Additional studies should look at the developer perspective. Would developers have the same view on the effectiveness and limits of the boundary objects, or would they identify different boundary objects? Would there be any differences between front-end or back-end developers?

Since the project and its constituent sub-projects had such a clear distinction between requirements gathering tracks and development tracks, some of the involved UCD specialists in the projects had to be excluded from the study as they had not been communicating with developers through boundary objects. This meant that only six UCD specialists participated in the study. This rather low number may have affected the validity of the study. But this is to some extent compensated as the UCD specialists participating came from several sub-projects and still expressed somewhat similar experiences.

The data sources in the study were limited to interviews. It would have been interesting to directly study the use of boundary objects through observation or document analysis. This could also reveal other boundary objects that were not mentioned by the participants.

Communication problems can be associated with cultural differences. Further studies should look more carefully at the relation and empathy between designers and developers in agile development. This can influence the developers' willingness to implement designs [23] and use design objects.

CONCLUSIONS

This study has investigated UCD specialists' view on boundary objects for communication with developers in agile projects. The purpose of an actual boundary object was mainly seen as communicating an understanding of the interaction with the system and not necessarily an understanding of the user. The boundary objects created for this purpose were neither considered to be comprehensive nor stand-alone.

With respect to the boundary objects themselves, the study has shown the importance of well-functioning boundary object repositories with easy access. Without such repositories, the objects risk losing their boundary qualities. Although hi-fi prototypes are time consuming in their initial creation, they could save time on maintenance and updating as the design process continues throughout the development. The study also showed that UCD specialists tended to adapt their usual way of working to the special needs of agile development. For example, they translated their work into deliverables such as user stories with associated design sketches. The reason for doing this was to help developers structure their work and connect the user to the design and its elements.

Concerning the communication of UCD specialists' understanding of the user with developers, three themes were identified. Although identified problems were seen to strongly communicate an understanding of the user, any documents and deliverables containing these problems were not received by developers. For developers to truly understand the user, they would have to participate to some degree in the user research. The alternative, as participants reported, was to verbally communicate their understanding of the user to developers on demand. This makes the connections between the design and the user internalized within each individual participant. However, higher-level connections become externalized by coupling user stories with sketches and/or prototypes.

ACKNOWLEDGEMENTS
We express our sincere gratitude to the participating UCD specialists, our champions at the usability consulting company, and the anonymous reviewers.

REFERENCES

1. Aberg, J. Challenges with Teaching HCI Early to Computer Students. *Proc. ITiCSE'10*, ACM Press (2010), 3-7.
2. Bayer, H. and Holzblatt, K. *Contextual design: Defining customer centered systems*. Morgan Kaugmann, 1998.
3. Brown, J., Lindgaard, G. and Biddle, R. Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artifacts. *Proc. Agile'08*, IEEE (2008), 39-50.
4. Blomkvist, S. Towards a model for Bridging Agile Development and User-Centered Design. In *Human-Centered Software Engineering – Integrating Usability in The Development Process*, Springer (2005), 219–244.
5. Chamberlain, S., Sharp, H. and Maiden, N. Towards a Framework for Integrating Agile Development and User-Centered Design. In *Extreme Programming and Agile Processes in Software Engineering*, Springer (2006), 143-153.
6. Cooper, A. *The inmates are running the asylum*. Macmillan, USA, 1999.
7. Convertino G., Mentis H.M., Rosson M.B., Slavkovic A. and Carroll J.M. Supporting content and process common ground in computer-supported teamwork. *Proc. CHI'09*, ACM Press (2009), 2339-2348.
8. Ferreira, J., Noble, J. and Biddle, R. Agile development iterations and UI design. *Proc. Agile'07*, IEEE (2007), 50-58.
9. Fox, D., Sillito, L. and Maurer, F. Agile Methods and User Centered Design: How These Two Methodologies Are Being Successfully Integrated in Industry. *Proc. Agile'08*, IEEE (2008), 63-72.
10. Hsieh, H-F. and Shannon, S.E. Three approaches to Qualitative Content Analysis. *Qual Health Res 15*, 9 (2005), 1277-1288.
11. Hussain, Z, Slany, W. and Holzinger, A. Investigating Agile User-Centered Design in Practice: A Grounded Theory Perspective. In *HCI and Usability for e-inclusion*, Springer (2009a), 279–289.
12. Hussain, Z., Slany, W. and Holzinger, A. Current State of Agile User-Centered Design: A Survey. In *HCI and Usability for e-inclusion*, Springer (2009b), 416–427.
13. ISO 9241-210:2010 Ergonomics of human-system interaction — Part 210: Human-centered design for interactive systems.
14. Jiménez, M., Piattini, M. and Vizcaíno, A. Challenges and Improvements in Distributed Software Development: A Systematic Review. *Advances in Software Engineering 2009*, (2009).
15. Larman, C. and Basili V.R. (2003) Iterative and Incremental Development: A brief history. *Computer 36*, 6 (2003), 47-56.
16. Larman, C. *Agile and iterative development: A manager's guide*. Pearson, 2003.
17. Lee, S. and Yong, H-S. Distributed agile: project management in a global environment. *Empir Software Eng 15*, (2010), 204-217.
18. Miller, L. Case Study of Customer Input for a Successful Product. *Proc. Agile'05*, IEEE (2005), 225-234.
19. Modi, S., Abbott, P. and Counsell, S. Negotiating Common Ground in Distributed Agile Development: A Case Study Perspective. *Proc. Int. Conf. Global Software Engineering*, IEEE (2013), 80–89.
20. Nielsen, J. *Usability Engineering*. Morgan Kaufmann, 1993.
21. Patton, J. Hitting the Target: Adding Interaction Design to Agile Software Development. *Proc. OOPSLA'02*, ACM Press (2002), 1–7.
22. Pruitt, J. and Adlin, T. *The persona lifecycle: Keeping people in mind throughout product design*. Morgan Kaufmann, 2006.
23. Redish, J., Bias, R.G., Bailey, R., Molich, R., Dumas, J. and Spool, J.M. Usability in Practice: Formative Usability Evaluations – Evolution and Revolution. *Proc. CHI'02*, ACM Press (2002), 885-890.
24. Salah, D., Paige, R.F. and Cairns, P. A Systematic Literature Review for Agile Development Processes and User Centered Design Integration. *Proc. Int. Conf. Evaluation and Assessment in Software Engineering*, ACM Press (2014).
25. Schwaber, K. *Agile Project Management with Scrum*. Microsoft Press, 2004.
26. Star, S.L. and Griesemer, J.R. Institutional Ecology, `Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. *Social Studies of Science 19*, 3 (1989), 387–420.
27. Sliger, M. and Broderick, S. *The software project manager's bridge to agility*. Addison-Wesley, 2008.
28. Sy, D. Adapting Usability investigations for Agile User-Centered Design. *Journal of Usability Studies 2*, 3 (2007), 112–132.
29. Ungar, J. The design studio: Interface design for agile teams. *Proc. Agile'08*, IEEE (2008), 519–524.