Agile-User Experience Design: Does the Involvement of Usability Experts Improve the Software Quality?

State of the Art and a First Experiment

Lou Schwartz
Luxembourg Institute of Science and Technology
5, avenue des Hauts-Fourneaux
L-4362 Esch/Alzette, Luxembourg
e-mail: lou.schwartz@list.lu

***Abstract*—In the past decade, numerous experiments and research proposed to take the advantages of Agile and User Centred Design methods in a mixed method called Agile-User Experience Design or Agile-UX. This combination raises a number of questions. Notably, it remains unclear who should be responsible of the usability in an Agile-UX project development. After a review of the literature on Agile, User Centred Design and Agile-UX, this paper focuses on the involvement of usability experts in Agile-UX. The literature discusses the involvement of usability experts in terms of processes and work methods, but never in terms of the necessity to involve usability experts to improve the software quality. To start answering this question, an experiment was conducted to explore the necessity to involve usability experts in the team. The results are that the involvement of a User Centred Design expert improves the quality of the developed product and the users' satisfaction in Agile-UX.**

Keywords-Agile; Agile-UX; User-Centred Design; Team composition; Involvement.

I. INTRODUCTION

Agile-User Experience Design (Agile-UX) is a project management principle for software development. It is based on Agile's values and principles, and on the User-Centred Design (UCD) method. Nowadays, no official definition of Agile-UX exists, but many experiments demonstrate its value [1][4][9][13][21][25][26][27][30][31][33][34].

Many questions still arise by this reconciliation of Agile and UCD. The one that this paper will deal with in depth is the necessity to involve a usability expert in the team. In the literature, Agile-UX is implemented with the involvement of usability experts in the Agile process and with the use of methods from UCD. But, in Agile, the intervention of experts is not encouraged [21] ("*UCD provides specialized skills in U[ser]I[nteraction] design but Agile approaches prefer generalists and discourage extensive upfront design work.*"). Rather, a dissemination of skills is preferred - by means of a "*generalizing specialist*" approach - to the intervention of experts [3]. This means it is preferable that team members can do all tasks to ensure a dissemination of the knowledge, including code knowledge, in the team and no one is left without work. Generalizing experts are multidisciplinary people able to work on different aspects or technics used in the project, like development and usability [3]. Furthermore, state of the art neither justifies nor discusses the involvement of usability experts in Agile-UX in term of necessity to improve the quality of the delivered software. In this paper, the involvement of usability experts in Agile-UX is discussed by testing both approaches within two experiments: the first one fully respects the principles of Agile project management: developers should be able to manage UCD themselves, and to conduct the related methods without the intervention of a usability expert; the second option integrates a usability expert in the project team to ensure both a better UCD implementation and results. We test three hypotheses: H1: without usability expert, if the project team has awareness and some knowledge in HCI, Agile-UX gives a correct quality level about the product's usability; H2: with usability expert involved in the project team, usability of the produced product is better than in H1; H3: the dynamic of the project team is better when a usability expert is involved.

After a reminder in Section II on the background composed of the definitions of the Agile method and the UCD method and their reconciliation in Agile-UX, a focus is placed on the literature review, with the particular research question on the involvement of usability experts in an Agile-UX development process in Section III. Afterwards, the paper presents an experiment in order to check our hypotheses in Section IV.

This paper is an extension of our contribution presented at ICSEA 2013 [1]; state of the art is extended and experiment' definition and results presentation are completed.

II. BACKGROUND

To better understand our question on usability expert involvement in Agile-UX, this paper first goes back on the Agile and UCD methods and on the issues and interests to reconcile them in a mixed method called Agile-UX.

A. Agile methods

The Agile methods are management methods for software development, which are based on an iterative development of software in order to better answer to changing requirements. According to Lindvall [20], Agile methods can be defined as iterative, incremental, self-organizing and emergent.

1) Values: The Agile methods aim at enhancing the value of the delivered product to satisfy the customer's requirements. The production is organized in iterations (or sprints) from two to eight weeks. Agile methods plebiscite the following four values defined in the Agile Manifesto [2]:

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

The Agile movement was instigated and pioneered by software developers in reaction to a frustrating history of projects being delayed, going over budget, collapsing under their own weight and stressful jobs. For the Agile manifesto founders, these problems have their origin in the excessive analysis, specifications and designs done before code writing that enabled unstable or not useful requirements and incompleteness. With the Agile methods, customers would obtain faster working software that better corresponds to their actual needs, thanks to the flexibility provided with the development process [4].

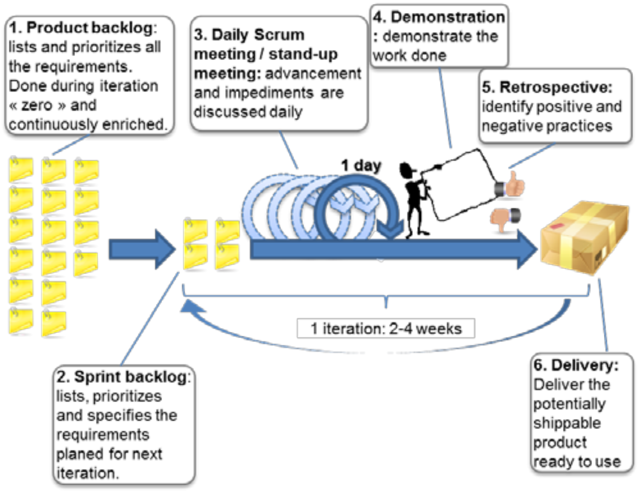
1. Product backlog: lists and prioritizes all the requirements. Done during iteration « zero » and continuously enriched.
3. Daily Scrum meeting / stand-up meeting: advancement and impediments are discussed daily
4. Demonstration : demonstrate the work done
5. Retrospective: identify positive and negative practices
1 day
1 iteration: 2-4 weeks
2. Sprint backlog: lists, prioritizes and specifies the requirements planed for next iteration.
6. Delivery: Deliver the potentially shippable product ready to use

Figure 1. Scrum process.

2) Most used methods: Today, the two most used Agile methods [12] are Scrum [28] (see Figure 1) and eXtreme Programming (XP) [6], or a mix of them, including the proposed integration of Agile methods and UCD [16]. Scrum focuses on management practices instead of development or software engineering practices [19]; it is then easier adaptable for the integration of other experts' practices like UCD. This certainly explains why it is the most used in reconciliation between Agile development methods and UCD.

3) Weaknesses of the method: some Agile methods are more focused on the developers' work and on the development quality, like XP. And even if the aim of Agile methods is to satisfy the product owner, they define neither method nor good practice to achieve this objective, particularly for the needs elicitation or the design part. The needs elicitation is done by the product owner, based on his proper knowledge of the domain or of the work done by users. He can use the methods he wants, including involvement of users (e.g., by interviews, context inquiries, etc.). After that, the needs are discussed within the team to refine and prioritize them, based on the business value but also on their technical complexity or on the necessity that previous work was done to realize them.

Concerning the UI design, it depends on the openness to the usefulness and usability of developers, the customer and the consulted users, so there is no guarantee about ergonomics [7]. Indeed, the product owner and developers are often not trained on the UCD approach and the associated methods. Developers are more focused on the client's needs than on the users' needs, and a lot of time and work are required from the product owner to gather all the user's needs and feedback. Unfortunately, the product owner is not only within that function, but often he continues to work on his normal tasks as employee of the organization. Thus, product owners do not have enough time for this additional task. It is the same for developers; they often have plenty of other tasks with only a limited timeframe left to set up a user centred approach. Moreover, Agile pushes developers, and often also the product owners, to focus only on a single set of functionalities (the user stories developed during the current iteration), so they sometimes lose the holistic view, which as a consequence, presents homogeneity problems. That is why it seems a good option to involve UCD experts to ensure staying in line with the real end-users needs, to organize the UCD approach, to implement the required UCD methods and to maintain a holistic view of the final software design.

The use of the UCD principles and methods is one way to ensure answering to users' needs. Based on these assessments, it seems that Agile teams can benefit from integration of UCD methods with Agile, in particular to improve the needs elicitation and the design part.

B. User-Centred Design

UCD focuses on producing usable software that not only satisfies real users' needs, but also those of customers. This method, described by the ISO 9241-210 standard [18] (see Figure 2), defines the process to follow in order to produce software that meets the users' requirements. It includes notably the design and the validation phases. By nature UCD is not focused on the developers' work.

1) Principles: The principles of the UCD are listed below [18]:

- The design is based upon an explicit understanding of users, tasks and environments.
- Users are involved throughout design and development.

- The design is driven and refined by user-centred evaluation.
- The process is iterative.
- The design addresses the whole user experience.
- The design team includes multidisciplinary skills and perspectives

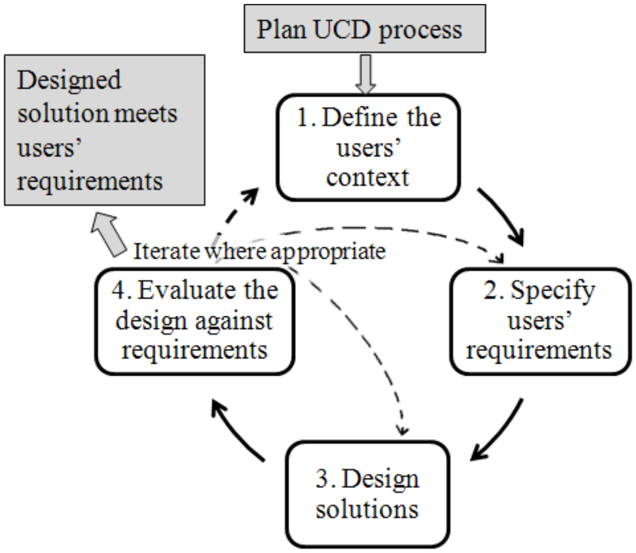
Plan UCD process
1. Define the users' context
2. Specify users' requirements
3. Design solutions
4. Evaluate the design against requirements
Designed solution meets users' requirements
Iterate where appropriate

Figure 2. UCD process as described by the ISO 9241-210 standard [18].

2) UCD methods: The implementation of the UCD process involves many methods (like prototyping, observations, interviews, users' tests, etc., see ISO/TR 16982:2002 [17] for descriptions of some of them) to support, amongst other things, the users-needs' definition and the validation of the delivered software by end-users. These methods are conducted by usability experts. They select the more appropriate methods concerning the context of the project (including constraints like budget and planning, the access to users, the available skills in the teams, etc.).

Agile and UCD processes are quality processes, which have the objective to provide the most suitable software with minimal issues. They are also both iterative. Then, they seem compatible and could enrich each other. In the next section, their compatibility will be discussed.

C. Reconciliation of Agile and UCD and research questions that arise

Even if some Agile concerns could prevent a UCD attitude [7] (focus is often more on programming techniques and programmers, automated tests, very short iterations, fast increments and executable software as a measure), a reconciliation of both approaches is possible and has often been implemented. Since a decade, several works propose to reconcile Agile and UCD [4][7][10][19][21][22][23][29] [33]. Several experiments indicate that an integration of Agile and UCD produces some interesting results [9][13][16][21][31]. As Nelson presents them, "*[XP (or Agile methods) and interaction design (or UCD) are] process[es] with similar goals but different methodology*. [23]" In fact, the two methods have a lot of compatibilities, but some impediments require adapting both to be efficient (see Table I for a synthetic view of conflicts and compatibilities between Agile and UCD). This reconciliation raises a lot of research questions. Some of them are listed in the following parts and, in the following section, the focus is placed on one particular question raised about the necessity of the involvement of UCD experts in Agile-UX.

1) Impediments to a mixed method: We can particularly note the following impediments and resulting questions.

In Agile methods, the intervention of experts is not encouraged and generalists or generalizing specialists are preferred [3][21]. UCD proposes the involvement of usability experts. So our questions are:

- Who should be in charge of UCD in an Agile project: team members or involved usability experts?
- How can usability experts be involved in the team?

In Agile, teams include a product owner, who is the customer and, de facto, the user representative [26]. In UCD this role is taken by a usability expert [21]. Our questions about this are:

- What are the responsibilities and activities of each role?
- It is necessary to keep these two roles?

Agile discourages extensive upfront design work while it is common that in UCD a deep analysis is done upfront [21]. So the questions are:

- Is it possible to reduce the first analysis done in UCD to fit the iteration duration and thus, realize this analysis during the iteration called *zero* [29]?
- Could the analysis be disseminated throughout the project?
- The deep analysis done upfront in UCD has the objective to provide a global vision and enable more homogeneity. How can a global vision and homogeneity be ensured?

UCD recommends the use of some design artefacts to facilitate communication with the project team, while Agile advocates focusing more on software developed than to produce unused documentation [9]. (Agile principle 2: Working software over comprehensive documentation [1]).

- Are some artefacts of UCD useful and simple enough to produce and understand, to improve communication without intruding upon the effort to produce working software?

The evaluation is done on different levels: from low-fidelity designs to software [26] in UCD with often only few users, and in Agile on production-ready application by real users in their real context [21].

- Are both levels of evaluation necessary?
- How to synchronize them?
- What are their specific objectives?

TABLE I. SYNTHETISIS OF CONFLICTS AND COMPATIBILITES IN AGILE AND UCD.

Agile	**UCD**	**Compatible**	**Practices proposed in Agile-UX to ensure compatibility**
Prefers generalists with some expert knowledge (generalizing specialists) Small team	Involvement of any kind of experts necessary for the project included UCD expert Multidisciplinary team	No	No involvement of UCD expert, but have someone with UCD knowledge in the team (developer, coach or product owner). Or - Involve when needed 1 UCD expert. Or - Involve 1 or several UCD experts throughout the project.
Product owner is customer and de facto user representative	UCD expert is user representative and de facto to a certain extent customer representative	No	UCD and product owner are necessarily involved in the project. Or - UCD expert is the product owner.
Agile discourages extensive upfront design work	In UCD often a deep analysis is done upfront	No	Reduce the upfront analysis, use the iteration *zero* to do it. Design the global vision (overall layout, navigation and look&feel) upfront, then detailed throughout the project when it is necessary and maintain the global vision. Or - Do a deep upfront analysis phase before an Agile development phase.
Value 2: Working software over comprehensive documentation	UCD recommends the use of some design artifacts to facilitate communication with the project team	No	Use only high value artifacts. And - Simplify the methods of artifacts' production. And - Simplify artifacts presentation. And - Produce the artifacts only when they are needed And - Disseminate their results at the end of each iteration Provide a visionary prototype of the final product maintained throughout the process
Evaluation is done on a production-ready application by real users in their real context	Evaluation is done on different levels: from low-fidelity designs to software, often with only few representative users	No	Both are complementary and needed
Agile is focused on code production and work of developers	UCD focuses on user interfaces and interactions and work of usability experts	No	Agile-UX proposes a reconciliation of these two points of view. Different processes are proposed to support the parallel work of UCD experts and developers
Focus on quality of the product. With regard to the Agile method used (XP, Scrum, Lean, etc.) and team skills, in Agile, the focus is placed either on the quality of the code or the quality of the product	Focus on usability and utility of the end product as measure of the quality of the product	Partially	Agile-UX requires taking into consideration factors such as utility and usability to ensure that the focus is well done on product quality
Satisfaying the customer	Concentrating on the user needs	No	Have a real end-users representative as product owner, support the product owner in the end-user needs identification and understanding thanks to the constant involvement of end-users throughout the development asked by UCD methods.
Iterative	Iterative	Yes	
Allow involvement of end-users and give access as soon as possible to working software to real end-user in their real context of user	Focus is done on end-users Feedback of end-users are essential	Yes	Can bring along more contextual and complete information to UCD experts as users' tests in laboratory or with all-comers users
Multi-disciplinary is not rejected by Agile, even if the involvement of experts has to be limited	Multi-disciplinary is a key value of UCD	Yes	Involvement of UCD experts in the Agile team when usability of the product is defined as a very important quality needed
Provides solid foundation for a user-centred attitude	User-centred attitude is a key value of UCD	Yes	
Continuous testing throughout the project of the developed software	The design is driven and refined by user-centred evaluations, and this is a key value of UCD	Yes	
Value 4: Responding to change over following a plan	Accept change: coming from users' feedback or from customers (context)	Yes	
Quality process of the code produced (bug free) and to ensure to answer the needs defined by the product owner	Quality process of the interfaces and interactions (usability) and ensuring to answer the real end-users' needs	Yes	Support the product owner on the definition of the users stories to ensure the representation of the real end-users' needs
Reduce costs of development by developping a bug free software and by the development of only the expressed needs (no more)	Reduce the cost by avoiding design errors that will reduce training time of users, avoid the rejection of software by users and decrease the risk of improving the developed software thereafter	Yes	

Agile is focused on valuable software production by ensuring the quality of the product. With regard to the Agile method used (XP, Scrum, Lean, etc.) and team skills, in Agile, the focus can be placed either on the quality of the code or the quality of the product. Regarding UCD, this focuses on user interfaces and interactions quality and work of usability experts [13][26]:

- Should the priority be given to the best practices and values of one (Agile or UCD) or both?
- What are the relations between developers and usability experts in Agile-UX?
- Do the usability experts and the developers find their place and feel well in Agile-UX?
- How to organize the development and the design work, what are the processes? Agile focuses on satisfying the customer, who is supposed to be a representative of the end-users and knows their needs. UCD focuses more on answering the user needs, while taking into account the overall context provided by the client organisation and their representatives [26].

2) Compatibilities that encourage a mixed method: Agile and UCD also have compatibilities.

Agile methods and UCD are both iterative processes even if the lengths of their cycles are different (some weeks in Agile, some months in UCD [21]).

- Can UCD and Agile cycles be synchronized?
- What are the different steps of each process and are they aligned? Agile methods allow an involvement of end-users and provide access to the working software the real end-users in their real context of use as soon as possible, which can bring along more contextual and complete information than users' tests in laboratory or with all-comers users. User feedback is also important in Agile development methods [19].
- How and when to involve users in the design and validation of the software?
- How to deal with the users' feedback?

Multi-disciplinary is not rejected by Agile, even if the involvement of experts has to be limited [21]. Indeed, as Blomkvist exposes, an Agile project culture provides a solid foundation for a user-centred attitude: "*a focus on people, communication, customer collaboration, adaptive processes and customer/user needs. [7]*"

- Continuous testing throughout the project is a good practice for both:
- How to integrate UCD tests with the Agile process and practices?
- What role do the UCD tests play in acceptance tests?
- Both accept change: coming from users' feedback or from customers [23].
- Finally, both are quality processes [14]: UCD aims to deliver quality software adapted to users, to their needs, to their context and to their tasks. Agile aims to deliver quality software without bugs and, which is adapted to the needs and constraints expressed by the product owner.
- Both have also the objective to reduce development costs: UCD by avoiding design errors, which will reduce training time of users, avoiding rejection of software by users and decreasing the risk of having to improve the developed software afterwards; and Agile by doing only what is asked and reducing the bug fixing after release [23].

3) Conclusion: To conclude, Agile methods do not cover all UCD's principles, but there is no blocking contradiction between Agile and UCD approaches and conversely. This certainly explains the increasing number of experiments or propositions of mixed methods.

In the next part of this paper, we will focus on a particular question raised in the literature review: the involvement or not of usability experts in the Agile-UX team. The next sections will deal with this question deeply through a more focused state of the art and the proposition of experiments to test the validity of our hypotheses.

Agile methods and UCD are both iterative processes even if the lengths of their cycles are different (some weeks in Agile, some months in UCD [21]).

III. Usability experts involvement in Agile-UX

An expert is defined in the Oxford dictionary as "A person who is very knowledgeable about or skilful in a particular area." Specialists or experts are professionals that have deep knowledge and skills concerning a particular domain, technology or methodology. They are focused on their subject of expertise.

Generalizing specialists are experts on several subjects. "They have one or more technical specialities but also a general knowledge in other areas of software development." Agile fosters this overspecialisation, by encouraging team workers to work on tasks outside their speciality [3].

UCD fosters an expert approach, while Agile rather advocates a generalizing specialist approach. How have these different approaches been mixed together in Agile UX? Through numerous experiments of Agile-UX, the question of *who is in charge of UCD* often comes up [4][9][13][15][21][23][25][30][31][33][34]. Different options are exposed, which can be grouped as explained below.

A. Specialist approach

One or more specialists (UCD experts) are involved in the Agile-UX process. The consistency of the interventions can be different from one project to another: from some punctual interventions (often at the beginning of the project, for conducting users' tests or on demand) to a constant presence throughout the project (often following Sy's parallel tracks process).

1) One usability expert: Only a few experiments advocate the integration of only one person in charge of UCD in Agile-UX (project 1 and project 3 in [13], P1, P5, P9 and P10 in [15], project PV in [21], [30][31]).

2) A parallel team of several usability experts: In most cases, a parallel team of several usability experts is dedicated to the project ([4][9], project 2 and project 4 in [13],[23][25][33], P2 and P4 in [15], [34]). Still, they organise the exchanges and the work between developers and designers differently.

B. *Generalizing specialist approach*

In a generalizing specialist approach, the product owner (project 1 in [13]) or some developers (project 3 in [13], P3, P6 and P8 in [15]) conduct also the UCD.

1) UCD expert as product owner: With regard to the UCD expert's and product owner's responsibilities, it is sometimes preferable to merge both roles (project 1 in [13], project TB in [21], defended by Beck in [23], [31][33]). The product owner, de facto, has a lot of responsibilities that can be taken into charge by UCD experts and UCD methods (see Table II) taking into consideration the UCD experts' role and responsibilities.

Furthermore, some observations show that the product owner is often overcharged with marketing and sales concerns. He often does not have the skills to manage a user-centred design, and, as a consequence, he may lose focus on a user experience vision [31].

TABLE II. HOW UCD EXPERTS CAN TAKE IN CHARGE PRODUCT OWNER RESPONSIBILITIES.

Product owner responsibilities [31][32]	**How UCD experts can take into charge product owner responsibilities**
Define the features of the product and decide on release date and content	UCD experts can define the user stories to develop based on gathered data on users, context and tasks [31].
Be responsible for the value of the product	By context studies, exchanges with the organisation on the needs and the attempted value, and observations of users and their feedback, UCD experts can define the value of the product and define priorities.
Prioritize features according to market value	
Can change features and priorities every 30 days	UCD expert accepts changes and modifies designs when it is necessary, based on users feedback. He can modify user stories and prioritizations according to new analysis.
Accept or reject work results	UCD experts use users' tests, expert validations to reject or accept the work results. These methods can be part of the acceptance tests.
Negotiate with all stakeholders	
Communicate with users and train them	This is also a responsibility of the UCD expert role.

TABLE III. WHO IS IN CHARGE OF UCD IN AGILE-UX, SYNTHETISIS OF THE LITERATURE REVIEW.

		Who is in charge of UCD in Agile-UX team?				**Consistency of the intervention**		**Good practice**
		Specialist approach		***Generalizing specialist approach***				
		1 usability expert	*Parallel team of UCD experts*	*1 UCD expert as product owner*	*UCD is done by developer(s) or other team member(s)*	***Punctual intervention***	***Constant presence throughout the project***	***Pair designing***
Armitage [4]			X					
Chamber-lain [9]	Project I		X					
	Project M		X					
	Project S		X					
Ferreira [13]	Project 1			X				
	Project 2		X					
	Project 3				X	X		
	Project 4		X					
McInerney [21]	[21]				X		X	
	Project MG					X		
	Project PV				X	X		
	Project TB			X	X			
Nelson [23]			X	X				
Nodder [25]			X					
Nummiaho [26]							X	X
Singh [31]				X				
Sy [33]			X	X			X	
Fox [15]	P1	X						
	P2		X		X			
	P3				X			
	P4		X		X			
	P5	X						
	P6				X			
	P7							
	P8				X			
	P9	X						
	P10	X						
Patton [27]					X			
Silva [30]		X					X	
Wale-Kolade [34]			X		X	X	X	

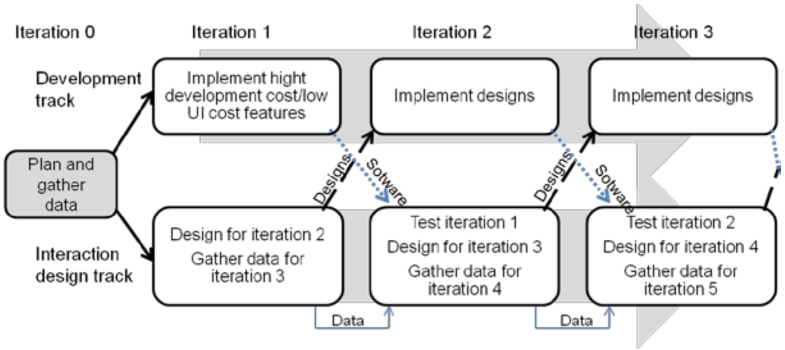

Figure 3. Parallel tracks of work of development and interaction design proposed by Sy [33].

2) Team members as responsible of the UCD process: The last possibility is to make some team members responsible for the UCD process. It is also the more close to the Agile visions: have team members who are generalizing specialists, who thus combine, for instance, skills in development and in UCD (project 3 in [13], P2, P3, P4, P6 and P8 in [15], [21][27][34]).

C. *Work organisation*

In addition to the question on the distribution of UCD responsibilities, the organization of UCD tasks is addressed in the literature.

Sy [33] proposed a parallel track organisation of work: designers work with one or two iterations ahead of developers (see Figure 3). To implement this proposition, several usability experts are needed, because of the amount of work. Several teams adopt this work organisation ([4][9], project 3 and 4 in [13], [26][34]), but sometimes with only one UCD expert [30].

In Agile methods, it is possible to dedicate a spike (an iteration to focus on a particular problem like testing a new technology) to usability exploration. Still, it is not a good solution to maintain a constant pace [25].

Some projects also occasionally involve UCD experts on some particular points (projects MG & PV in [21], [34]); this is close to an organisation by spikes. But, for McInerney [21], it is important that the usability expert is available *on call* at all times, which may be impossible if the usability expert works on several projects simultaneously.

Some other projects integrate usability in the iteration without real planning (see [P3.290] in [13]).

D. *Synthesis*

In the literature, both modalities can be founded: involving or not a UCD expert. We can note a preference for the involvement of UCD expert(s).

To summarize, in literature, Agile-UX teams involve at least one UCD expert most of the time (see Table III). His role, the consistency of his intervention, and the synchronization of his intervention with the developers' work are not fixed, even if the parallel tracks of design and development seem to be the most adopted practice. In the studied papers, there is no mention of why a UCD expert in Agile-UX should be involved. There is no reference on the fact that it could or not be better to involve a UCD expert in the team. This can raise the following question: is it necessary to involve usability experts in the team, or is involving team members with some knowledge on usability sufficient? This is what we tested in the implemented experiment presented in the following section.

IV. EXPERIMENT

After the literature review and interviews with Agile professionals, we focused on the question of the usability expert involvement in the team. Literature contains a lot of experiments on the involvement of usability experts in an Agile team, but they are more focused on the process and methods used than on the necessity to involve usability experts in Agile-UX. Based on this observation, we propose the following hypotheses to check the importance to involve usability experts in Agile-UX team in order to improve the produced software's quality in terms of usability:

H1: without usability expert, and if the project team has awareness and some knowledge in HCI, Agile-UX renders a correct quality level about the product's usability.

H2: with usability expert involved in the project team, the usability of the produced product is better than in H1.

H3: the dynamic of the project team is better when a usability expert is involved.

We retrospectively and qualitatively question these hypotheses through an experiment. The focus is done only on the usability of the final product, laying aside any potential overhead costs induced by the involvement of a UCD expert.

A. *Context of the experiment*

The method used consists in a retrospective and qualitative analysis of two software development projects: the first one without a usability expert in the team (to challenge hypotheses H1 and H3), the second one with one UCD expert in the team (to challenge hypotheses H2 and H3). The observations will help us to better define the issues related to *who should take the responsibility of the usability expert in Agile-UX*? Both observed projects are instantiations of Agile-UX.

They aim to develop a mobile application prototype, in order to demonstrate the interest of mobile touch-based applications for the construction of site-related activities. The implemented prototype allows taking photos located on a construction site via a Global Positioning System (GPS. The user can highlight parts of a photo (e.g., add an arrow on a wall defect) and add textual or vocal notes about the entire photo or about the highlighted parts on the photo. The user can also register some construction sites by indicating their location on a map. Then, the photos are automatically attached to a construction site according to their location. The user can also find his photos in his calendar since the photos are automatically attached to events in his Google® calendar based on the shooting date. Finally, the user can share a set of photos with additional comments.

Two development projects were planned to experiment two different implementations of Agile-UX. Scrum was chosen as Agile method for both.

We chose small teams to facilitate this first observation and pay better attention to who does what, and what are the dynamics in the teams. The iteration durations were chosen by each team, according to the time deemed necessary by them to work at a convenient pace. The parallel tracks process from Sy was chosen as process in the second experiment because it was already used by a part of the team. We let the team choose the UCD methods they needed and the way to implement them (when and how). Dynamics throughout the projects have been observed, but usability of developed software was measured only at the end of the projects, in order to do not introduce a bias (e.g., like a competition between the teams).

1) Case #1 – Agile-UX without UCD expert: the first case studied does not involve a usability expert, so UCD is done by the team and particularly by the developer.

a) Composition of the team: In the first case, the team was composed of: a full-time developer, a Scrum master (part-time) and a business expert (part-time) who plays the role of product owner, and who is a researcher and an architect, with knowledge of architects' practices in France and Luxembourg.

All members of the team are aware to and have some knowledge in Human Computer Interaction (HCI) thanks to either an initial education that included courses on HCI or a business expertise.

b) Implementation of the UCD: The first case lasted six months with iterations' duration of one week. The team implemented Agile-UX on 22 iterations.

2) Case #2 - Agile-UX involving a usability expert: the second case studied involves a usability expert in the team, who is in charge of the deployment of the needed UCD methods.

a) Composition of the team: During the second case, the team was composed of a full-time usability expert (with an initial education on psychology and ergonomics), a full-time developer, a business expert (part-time) as product owner, and a Scrum master (part-time). The business expert and the Scrum master have either an initial education that included courses on HCI or a business expertise. The developer has neither particular awarness nor initial knowledge in HCI and, of course, he did not participate in case #1.

c) Implementation of the UCD: This development lasted six months with iterations of two weeks. Due to calendar constraints, the developer first started to work on technical requirements one month before the usability expert. For independent reasons, the usability expert quit the project before the end of the six months. The complete team only worked together for two and a half months. The process followed was the parallel track proposed by Sy [33].

B. Evaluation method

Different collecting methods and measure variables have been used to compare both projects, in order to challenge our hypotheses. They are presented below.

1) Quality of the product: The quality of the product developed by each team was measured with a focus on usability. The usability of each project has been measured by identifying the usability issues met by users, their number and their importance, but also by measuring the satisfaction of users.

Usability issues: Usability issues are problems encountered by users during the use of the software. For instance, they do not find the right button to perform an action, they are lost, some functionalities are missing, etc. Usability issues are raised thanks to users' tests. During users' test, users were asked to realize some scenarios, like find all photos taken during the last meeting. Errors made by users and their comments are observed, and are reformulated as usability issues. To define the importance of these usability issues, we propose the following formula see (1). Importance (i) represents the number of users who encountered the problem (n), multiplied by a seriousness indicator (s) stating whether a user was not able to pursue the interaction (from maximum level 4) or whether it was just a detail that did not impede the interaction (to minimum level 1).

$$i = n * s. \tag{1}$$

To evaluate the seriousness (or resolve's priority) of a usability problem, a decision tree inspired by the one defined by Cooper and Harper [11] was used. The decision tree (see Figure 4) enables to evaluate a seriousness level taking into account the importance of errors, their frequency and the users' success in fulfilling their tasks relative to the defined scenario.

Thus, to compare the quality of product of both projects, the number of problems met by users and the importance of problems raised were measured.

To complete this measure, we measure also the users' satisfaction.

a) Users' satisfaction: The users' satisfaction is defined as *"the users' success in fulfilling their tasks relative to the scenario"* in [18]. Several questionnaires exist to measure this satisfaction. We decided to use the

SUS questionnaire that was known by at least one person in each project team [8]. The questionnaire enables to calculate a percentage of satisfaction by user when using the software. The questionnaire was distributed to users at the end of their users' test.

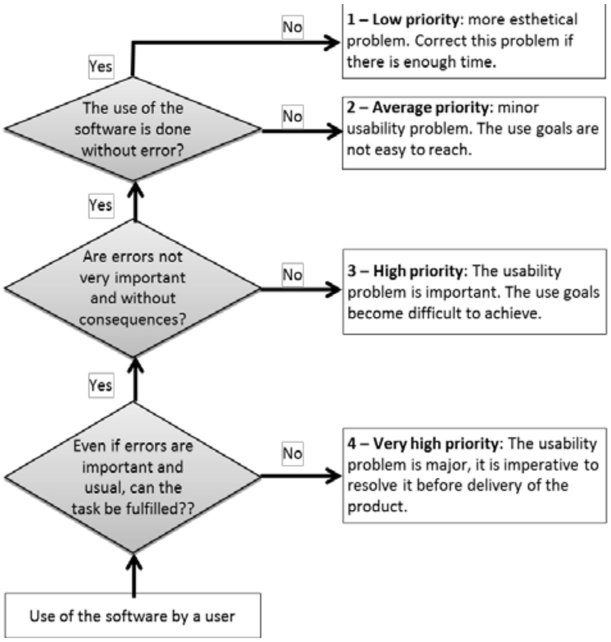
1 – Low priority: more esthetical problem. Correct this problem if there is enough time.
No
Yes
The use of the software is done without error?
No
2 – Average priority: minor usability problem. The use goals are not easy to reach.
Yes
Are errors not very important and without consequences?
No
3 – High priority: The usability problem is important. The use goals become difficult to achieve.
Yes
Even if errors are important and usual, can the task be fulfilled??
No
4 – Very high priority: The usability problem is major, it is imperative to resolve it before delivery of the product.
Use of the software by a user

Figure 4. Decision tree to evaluate the seriousness (s) of a usability problem.

2) *Team dynamics and satisfaction:* Team dynamics and satisfaction of the teams were observe thanks to interviews of each team member throughout the projects. We participated to all meetings of each team and we observed occasionally some work sessions.

C. Results

Results of the experiment are discussed in terms of the quality of the final product, with a focus on the usability, the implemented UCD methods and the observed team dynamics. Usability is defined in ISO 9241-210 as *"[usability is the] extent to which a system, product or service can be used by specified users to achieve goals with effectiveness, efficiency and satisfaction in a specified context of use.* [18]" Some screen shots of the application developed in the first and second project are shown below (see Figure 5 and Figure 6).

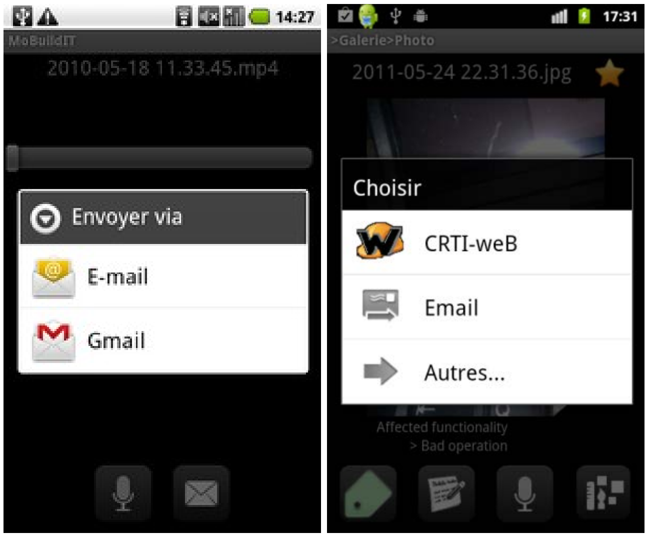

Figure 5. Photo sharing in first (left) and second project (right).

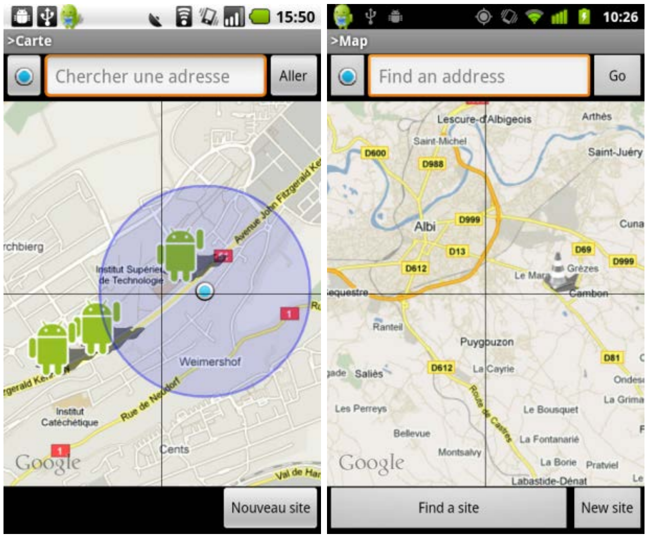

Figure 6. Site location in first (left) and second project (right).

1) *Quality of the final product:* challenging hypotheses H1 and H2, the quality of the product produced by each team with a focus on usability has been measured.

The users' tests raised fifteen problems encountered by users in the first case and only seven problems in the second one (see Table IV). Furthermore, problems are more important in the first case (11 problems with an importance between 1 and 8, and 4 problems with an importance between 10 and 20) than in the second one (7 problems with importance between 1 and 8 and no problem with importance higher than 8).

TABLE IV. USERS' TESTS RESULTS IN BOTH PROJECTS.

		Number of problems met	
		Use case 1	***Use case 2***
Importance of the problems (*i*)	1	5	2
	2	2	1
	3	3	1
	4	0	1
	6	1	1
	8	0	1
	10	1	0
	12	1	0
	15	1	0
	20	1	0
TOTAL		15	7

Users' satisfaction was measured thanks to the SUS questionnaire answered by users who made the users' tests [8]. It shows a lower average satisfaction in the first case (75.42%) than in the second one (81.25%). It can also be noted that extreme values are lower in the first case than in the second one (see Table V).

TABLE V. USERS' SATISFACTION RESULTS.

Percentage of users' satisfaction	**Use case 1**	**Use case 2**
Average	75.42 %	81.25%
Min	62.5 %	75%
Max	90 %	92.5%
Median	75	78,75
Deviation	8,74	6,73

2) *Team dynamics and satisfaction:* We used direct observations and interviews with team members in order to obtain a qualitative feedback on the team dynamics and satisfaction of team members on the interactions in the team.

Case #1: During this project, the developer played the role of UCD expert and developer of the application. As the developer had to play both roles, he had the feeling to progress slowly. Moreover, it is not easy to evaluate one's own work and to question it.

It should be noted that the team was in constant contact with the product owner thanks to his presence at every specification meeting, every demonstration meeting, and during some stand-up meetings. The product owner was also available to answer any team member's questions when necessary.

Case #2: The whole team had the feeling to progress quickly and to implement more functionalities, but also to obtain a better quality of the application.

Moreover, we observed the natural establishment of a *pair designing* [7][26]: when the developer was implementing wireframes, he sometimes asked the usability expert to join him and to explain and validate the developed interfaces during the implementation; when the usability expert designed wireframes, she sometimes asked the developer to join her and to validate the feasibility of wireframes during their design. Even if the developer had no skill in HCI at the beginning, he learnt the good practices throughout the project and quickly integrated them.

Furthermore, the team was in constant contact with the product owner through the Agile's dedicated meetings and also on demand.

3) *Methods used:* We also observed the UCD methods used in both projects. More UCD methods were deployed in the second case than in the first one, and both teams used different methods.

a) *Case #1:* The developer implemented only four usability methods: brainstorming, wireframing, users' tests, and satisfaction questionnaire. The following methods have been implemented by the team:

- Brainstorming sessions including business experts and technical experts to build the product backlog.
- Wire framing with Microsoft Power Point®.
- Two users' tests:
    - Real context of use, one user, one week
    - 6 architects, 6 scenarios, observation tests in laboratory. The results of these users' tests are presented in the previous section in Table IV.

b) *Case #2:* The following methods have been implemented by the UCD expert:

- Brainstorming sessions including business experts and technical experts to build the first version of the product backlog.
- Personas that help to define needs more precisely and improve the product backlog.
- Wire framing using paper & pen or sometimes Balsamiq®.
- Expert review based on ergonomics criteria (e.g., 10 usability heuristics for user interface design of Nielsen [24] or ergonomic criteria for the evaluation of human-computer interfaces of Bastien and Scapin [5]) after each release.
- Users' tests with four users: two users who know the application, and two novices. The results of these user tests are presented in the previous section, in Table IV.
- Focus groups to evaluate wire framing.

D. *Discussion*

Both projects' contexts and their results are synthesized in the Table VI. In the following, the hypotheses are discussed with regard to this experiment' results. Even if the results can be hardly generalized because of the very small sample size, our hypotheses tend to be confirmed.

1) *Hypothesis 1:* Results of users' tests and satisfaction questionnaire in case number 1 show that the users had encountered some problems. These problems are not very numerous (17) and their importance is relatively low (11 of the 17 problems noted have an importance inferior or equal to 8, with a maximum importance of 20). Satisfaction is quite good with an average satisfaction of 75.42%. With

regard to the first experiment results, Agile-UX works without a usability expert when some awarness and knowledge in HCI are available in the team. This justifies our first hypothesis H1.

TABLE VI. COMPARATIVE TABLE OF BOTH PROJECTS.

		Use case 1	**Use case 2**
Team	Developer	1 full-time	1 full-time
	Scrum master	1 part-time	1 part-time
	Product owner	1 part-time, business expert	1 part-time, business expert
	Usability expert	No	1 full-time
	Awarness to UCD	All team members	All team members, except the developer
Organisa-tion of work	Duration	6 months	Expected 6 months – but 2,5 months
	Iteration duration	1 week	2 weeks
	Number of iterations	22	5
	Process	Scrum	Scrum + Sy's parallel tracks
UCD methods	Wire framing	Power Point®	Paper and pen + Balsamiq®
	Users' tests in direct observation	6 users, 6 scenarios	At every iteration end with 2 users who know the application and 2 novices
	Users' tests in real situation	1 user during 1 week	No
	Satisfaction questionnaire	Yes: SUS	Yes: SUS
	Personas	No	Yes
	Expert review	No	Yes, with Bastien and Scapin criteria
	Focus groups	No	To evaluate the wireframes
Other methods used	Brainstorming	To build the product backlog	To build the product backlog
Team dynamic and satisfacti on	Feelings of the team	Slow progression	• Quick progression • Go further in the functionalities proposed • Improve quality of the application
	Observed team dynamic	No real dynamic Discouragement	• Pair-designing • Developer increased his HCI skills
Results		A lot of usability issues but working software.	• Lower number of usability issues identified by users and they are less critical. • Better users' satisfaction • And working software.

2) Hypothesis 2: We can detect that HCI skills of all team members would help avoid some usability mistakes in case 1, but, as the test results have shown, usability issues were identified by users. The test results show a lower number of usability issues in the second case (7 usability issues in the second experiment instead of 15 usability issues in the first experiment), thanks to the integration of the usability expert. It can also be noted that the usability issues are less critical in the second case than in the first case (see Table IV). Furthermore, the satisfaction of users, which is *correct* (75.42%) in the first case, is better in the second one (81.25%, see Table V). This justifies our second hypothesis H2: Agile-UX provides better quality in terms of usability with the involvement of a usability expert. This could also be explained by the different number and types of UCD methods used in both cases. In fact, in case 2, more methods were used because the usability expert was better trained and knew a wider variety of methods, but also because she had more time to dedicate to the deployment of these methods. Thus, an involvement of a usability expert in Agile-UX enables to use more methods, maybe more adapted, certainly best mastered.

3) Hypothesis 3: Without involving a usability expert, we observe a discouragement, particularly of the developer. On the contrary, involving a usability expert helps to maintain a constant pace in the team (principle 8 in [2]). No difference has been observed on the constant customer collaboration (value 3 in [1]). Some best practices emerged in the second case like *pair-designing*, and the whole team improved their practices and knowledge concerning HCI. This could justify our third hypothesis H3: the dynamic in the project team is better with a usability expert involved in Agile-UX. However, the fact that in the first case, the team was composed of only one person (the developer) may be of influence. Indeed, in the second case the team was composed of two people (the usability expert and the developer). Then, the dynamic observed may be due to the edge effect of the number of people in the team or simply to the personality of the people involved.

V. CONCLUSION AND FUTURE WORK

The literature review shows that the reconciliation of Agile and UCD is not a new trend, and it also shows that a number of research questions arise that have not been resolved today.

In this paper, we have further investigated the question of the necessity to involve a UCD expert in an Agile-UX team to support the UCD process. The state of the art shows that different types of involvement of UCD experts have been tried through different use cases, but the necessity of their involvement is neither justified nor discussed, and past experiments do not state the quality improvement implied by the involvement of UCD experts. To discuss this point, we proposed an experiment.

This experiment addressed two kinds of Agile-UX implementations. The first project did not involve a UCD expert in the team, but a team member, who was a generalizing specialist on development and on UCD was in charge of UCD. The second project involved a UCD expert in the team. With the help of these projects' observation, our two first initial hypotheses have been checked (H1-without usability expert, if the project team has awareness and some knowledge in HCI, Agile-UX gives a correct quality level about the product's usability; and H2-with usability expert involved in the project team, usability of the produced product is better than in H1). The third one cannot be checked at this step, even though observation shows that this hypothesis seems true (H3-the dynamic of the project team is better when a usability expert is involved). Further studies have to be conducted to have more quantitative results and to check the third hypothesis. Notably, the experiment protocol can be improved by evaluating also the usability of the developed software thanks to objective criteria like Nielsen usability heuristics [24] or Bastien and Scapin ergonomics criteria [5]. To that end, software are reviewed by usability experts thanks to one criteria guide. Usability issues can be grouped by kind of non-respecting criteria. An importance can be also calculated for each issue as we did for usability issues met by users during users' tests. During the experiment presented in this paper, all latitude was let to teams to select the UCD methods to use. The team 1 did not use this kind of expert review driven by objective criteria. Whereas team 2 chose to conduct at least one expert evaluation. It would be interesting to do a final expert evaluation to better compare objectively both results.

Future experiments will enable to measure the quality, in terms of usability, of software developed in Agile-UX. Completed projects that cover the different following variations will be selected:

- Number of UCD experts involved in the project, from zero to as many as there are developers involved in the team.
- Involvement modality of the person in charge of UCD: as a team member, as an external provider of services or as the product owner.

For each couple of parameters, at least ten projects with larger teams than the experiment presented above will be selected. The usability of selected completed projects will be measured through users' tests, satisfaction questionnaire and final heuristic evaluations driven by objective criteria (e.g., Nielsen criteria [24] or Bastien and Scapin ergonomics criteria [5]). These results will enable us to conclude what is the involvement modality that produces more usable software. In addition, the UCD methods deployed, their implementation mode (when and how), and the organization of work between developers and people in charge of UCD will be observed. Such observations would enable to identify good practices.

Another possible implementation of Agile-UX, which can be found in literature, is to place the usability expert as the product owner. In fact, the product owner is responsible for the contact with users, the definition of needs and the validation of the work done. A priori, some of the high level responsibilities of both, the usability expert and the product owner, overlap. A future task will be to check the legitimacy of the following hypothesis: the UCD expert could play the role of product owner.

ACKNOWLEDGMENTS

The author thanks members of the CRAI laboratory of Nancy and the project's extended team members: Sylvain Kubicki, Annie Guerriero, Fabrice Absil, Marion Zéler, Luc Caffard and Charles Gilbertz. The author also would like to thank Jocelyn Aubert, Muriel Foulonneau. This paper is dedicated to Marion Zéler.

REFERENCES

[1] L. Schwartz, "Agile-user experience design: with or without a usability expert in the team?," In Proc. ICSEA 2013, pp. 359-363.

[2] A. Alliance, "Agile manifesto," 2001, http://www.agilemanifesto.org, last access 2014.11.12.

[3] S. Ambler and M. Lines, "Disciplined agile delivery: A practitioner's guide to agile software delivery in the enterprise," 2012, IBM Press.

[4] J. Armitage, "Are Agile methods good for design?," Interactions, vol. 11, no 1, 2004, pp. 14-23.

[5] C. Bastien and D. Scapin. "Ergonomic criteria for the evaluation of human-computer interfaces," RT-0156, 1993, p.79.

[6] K. Beck, "Extreme Programming Explained: Embrace Change," Addison-Wesley Professional, 2003, p. 54.

[7] S. Blomkvist, "Towards a model for bridging Agile development and user-centered design," In Human-Centered Software Engineering—Integrating Usability in the Software Development Lifecycle, 2005, pp. 219-244, Springer Netherlands.

[8] J. Brooke, "SUS-A quick and dirty usability scale. Usability evaluation in industry," 1996, vol. 189, p.194.

[9] S. Chamberlain, H. Sharp, and N. Maiden, "Towards a framework for integrating Agile development and user-centred design," In Extreme Programming and Agile Processes in Software Engineering, Springer Berlin Heidelberg, Oulu, Finland, June 2006, pp. 143-153.

[10] L. Constantine, "Process agility and software usability: toward lightweight usage-centered design," Software, IEEE, vol 19, no. 2, 2002, pp. 42-50.

[11] G. E. Cooper and R. P. Harper, "The use of pilot rating in the evaluation of aircraft handling qualities," No. Agard-567, Advisory group for aerospace research and development, Neuilly-Sur-Seine, FRANCE, 1969.

[12] T. Dybå and T. Dingsøyr, "Empirical studies of Agile software development: A systematic review," Information and software technology, 2008, vol. 50, no. 9, pp. 833-859.

[13] J. Ferreira, J. Noble, and R. Biddle, "Agile development iterations and UI design," In Agile Conference (AGILE), Washington, DC, August 2007, pp. 50-58, IEEE.

[14] J. Ferreira, H. Sharp, and H. Robinson, "User experience design and Agile development: managing cooperation through articulation work," Software: Practice and Experience, vol. 41, no. 9, Wiley Online Library, 2011, pp. 963-974.

[15] D. Fox, J. Sillito, and F. Maurer, "Agile Methods and User-Centered Design: How These Two Methodologies Are Being Successfully Integrated In Industry," In Proc. Agile 2008, IEEE, 2008, pp. 63-72.

[16] J. Haikara, "Usability in Agile Software Development: Extending the Interaction Design Process with Personas Approach," Agile Processes in Software Engineering and Extreme Programming, Springer, 2007, pp. 153-156.

[17] ISO 16982:2002, "Ergonomics of human-system interaction -- Usability methods supporting human-centred design," International Standards for Business, Government and Society.

[18] ISO 9241-210:2010, "Ergonomics of human-system interaction -- Part 210: Human-centred design for interactive systems," International Standards for Business, Government and Society.

[19] D. Kane, "Finding a place for discount usability engineering in Agile development: throwing down the gauntlet," in Proceedings of the Agile development conference, 2003, pp. 40-46.

[20] M. Lindvall et al., "Empirical findings in Agile methods, Extreme Programming and Agile Methods," XP/Agile Universe 2002, Springer, 2002, pp. 81-92.

[21] P. McInerney and F. Maurer, "UCD in Agile projects: dream team or odd couple?," Interactions, vol. 12, no. 6, 2005, pp. 19-23.

[22] L. Miller, "Case Study of Customer Input For a Successful Product," in Proceedings of the Agile Development Conference, July 2005, pp.225-234.

[23] E. Nelson, "Extreme programming vs. interaction design," FTP Online, 2002.

[24] J. Nielsen, "Heuristic evaluation," In Nielsen, J. and Mack, R.L. (Eds.), Usability inspection methods, 1994, vol. 17, no. 1, pp. 25-62.

[25] C. Nodder and J. Nielsen, "Agile usability: best practices for user experience on Agile development projects," Nielsen Norman Group, 2010.

[26] A. Nummiaho, "User-Centered Design and Extreme Programming," In Software Engineering Seminar, Fall 2006, pp. 1-5.

[27] J. Patton, "Hitting the target: adding interaction design to agile software develop," In OOPSLA 2002 Practitioners Reports, 2002, ACM.

[28] D. Rawsthorne and D. Shimp, "Scrum In A Nutshell," SCRUM alliance, http://www.scrumalliance.org/articles/151-scrum-in-a-nutshell, last access 2014.11.12.

[29] L. Schwartz, "Agile-User Experience Design: an Agile and User-Centered Process?," In ICSEA 2013, October 2013, pp. 346-351.

[30] T. S. D. Silva, M. S. Silveira, and F. Maurer, (2013, August). "Ten Lessons Learned from Integrating Interaction Design and Agile Development," In Proc. AGILE'13, Nashville, TN, August 2013, pp. 42-49, IEEE.

[31] LM. Singh, "U-SCRUM: An Agile methodology for promoting usability," In Proc. AGILE'08, Tonronto, ON, August 2011, pp. 555-560, IEEE.

[32] J. Sutherland and K. Schwaber, Scrum, "The scrum papers: Nuts, bolts, and origins of an Agile process," 2007.

[33] D. Sy, "Adapting usability investigations for Agile user-centered design," Journal of usability Studies, vol. 2, no. 3, 2007, pp. 112-132.

[34] A. Y. Wale-Kolade, P. A. Nielsen, and T. Päivärinta, (2014). "Integrating Usability Practices into Agile Development: A Case Study," In V. Strahonja, N. Vrček, D. Plantak Vukovac, C. Barry, M. Lang, H. Linger, & C. Schneider (Eds.), Information Systems Development: Transforming Organisations and Society through Information Systems (ISD2014 Proceedings), Croatia, September 2014.