UX information in the daily work of an agile team: A distributed cognition analysis

Luciana A. M. Zaina(corresponding) [a], Helen Sharp[b], Leonor Barroca[b]

[a]*Computer Department – Federal University of São Carlos (UFSCar), Rodovia João Leme dos Santos (SP-264), Km 110, Sorocaba-SP, Brazil*

[b]*Faculty of Science, Technology, Engineering and Mathematics – The Open University, Walton Hall,MK7 6AA, Milton Keynes, UK.*

Abstract

The integration of agile software development and user experience (UX) design has been a topic of investigation for practitioners and researchers for many years, and agile teams have become increasingly aware of the importance of UX design. Most studies have focused so far on the integration of UX theories and methods with agile practices. The objective of this research is to investigate whether and how UX information is embedded in the daily work of an agile team. We conducted an ethnographic study of an agile team based in the UK. We performed a qualitative analysis using different data sources and three complementary analytical lenses: Distributed Cognition of Teamwork, Garrett's set of UX elements and planes, and Hassenzahl's content-oriented model of UX. This combination provided an understanding of the different types of UX information available to the agile team through artefacts and face-to-face meetings, how the information flowed within and around the agile team, and the type of engagement they have with UX information. The findings reveal that: (1) agile team members were consumers of UX information not producers; (2) the most common type of UX information found in the system related to how the user interacts with the product rather than to user goals or needs; (3) information focusing on the user perspective appears in verbal communication rather than being captured in artefacts; and, (4) the flow of UX information around the team is complex. In combination, these factors indicate a potential breakdown in the communication of UX information. We argue that these findings have relevance for other agile teams because the artefacts and methods used by this team are commonly used by other agile teams. To improve the situation, we suggest a number of recommendations to engage agile team members in UX work, and reduce the complexity of UX information flow.

Keywords: user experience/UX information, agile team, cognitive system, information flow, DiCoT, distributed cognition, communication.

1. Introduction

User experience (UX) designers and agile developers have worked together to incorporate UX in agile projects for many years (Brhel et al., 2015; da Silva et al., 2011; Chamberlain et al., 2006). Integrating methods from these two areas presents several challenges. While agile practices motivate teams to deliver small feature sets of working software in tight iterations, UX design usually requires upfront research and analysis prior to development (da Silva et al., 2018; Kuusinen, 2015; Jurca et al., 2014; Fox et al., 2008).

Recently, agile developers have started to recognise that UX is the responsibility of the whole team and not only of UX experts (Ardito et al., 2014; Bordin and Angeli, 2016; McInerney, 2017). However, there are still unsolved problems, such as: sharing UX work with the whole team on an ongoing basis; traceability of UX information; and, the industrial impact of UX models and tools (Kashfi et al., 2017). These problems result in UX designers and agile developers running their activities in parallel, and developer teams having difficulty in seeing what UX work is done as UX is not explicit within agile activities (Cajander et al., 2013; Ferreira et al., 2012).

Typically, in agile development, communication takes place face-to-face with participants aiming to attain shared understanding (Abdullah et al., 2010), however communication about UX relies traditionally on the use of artefacts (Bordin and Angeli, 2016; Garcia et al., 2017). Also, knowledge of UX is often based more on experience and less on the application of UX models, theories or tools (Kashfi et al., 2017; Gray, 2016). The difficulties in using UX methods and tools have led agile developers to adapt artefacts they are familiar with (e.g. user stories), to support UX activities (Choma et al., 2016; Schon et al., 2017).

To the best of our knowledge, little is known about the characteristics of UX information that agile teams process, and how this information is used. Many of the studies on agile and UX focus on the integration of methods and processes from agile and UX (Wale-Kolade et al., 2013; Jurca et al., 2014; Brown et al., 2011), and on communication challenges (Ferreira et al., 2012; Bordin and Angeli, 2016; Garcia et al., 2019). Others present results based on data collected with developers through interviews, questionnaires and focus groups about agile and UX (Larusdottir et al., 2017; Ardito et al., 2014; Kashfi et al., 2017; Gray, 2016), but do not examine how agile teams process UX information in their daily work. Although some researchers observe individuals in their workplace (Bordin and Angeli, 2016; Ferreira et al., 2012) they do not characterise UX information nor explore how it is integrated in software development work. Motivated by the above, we formulated two research questions (RQ):

RQ1 - Which type of UX information do agile teams process in their daily work?

RQ2 - How does UX information flow within the cognitive system of an agile team?

To answer the RQs, we carried out an ethnographic study for 3 months in 2018 to examine the cognitive system of the daily work of an agile team supporting and developing a Virtual Learning Environment used by many thousands of students a year. We collected qualitative data through observation sessions of daily work and meetings, individual interviews and participation in demo sessions and a process workshop. We followed ethnographic principles to investigate the participants' perspectives rather than formulating pre-conceptions about UX. Our investigation sought to observe how UX information featured and flowed in the day to day activity of the team. Our findings show that: (1) agile team members were consumers of UX information not producers; (2) the most common type of UX information found in the system related to how the user interacts with the product, rather than to user goals or needs; (3) information focusing on the user perspective appears in verbal communication rather than being captured in artefacts; and (4) the flow of UX information around the team is complex. In combination, these factors indicate a potential breakdown in the communication of UX information for this team. We argue that these findings have relevance for other agile teams, as literature shows that the artefacts and methods used for UX by this team are commonly used by other agile teams. We identify three areas for improvement: making more visible to team members both the motivations that lead users to interact with a product, and the functionalities offered by a product to its users; increasing the level of engagement of team members with UX information; and, decreasing the complexity, for team members, of identifying and collating UX information

Our study builds on previous work but enhances it by bringing a perspective that focuses on the *type* of UX information available to the team, and a distributed cognition view of UX information propagation and transformation. This perspective supports the identification of potential communication breakdowns in information flow, and highlights the kind of UX information that is lacking in agile team working.

This paper is organised as follows. Section 2 presents related work, and the theoretical underpinnings that guided the ethnographic study. In Section 3, we explain the research method, describe the study context, its planning, data collection and analysis approaches. The analysis and findings focusing on the use of UX information are presented in Section 4. In Section 5, the research questions are considered and recommendations are presented. In Section 6, we discuss the study validity. Finally, Section 7 offers conclusions and suggestions for future work.

2. Background

2.1. UX and agile

User-centred design (UCD) consists of techniques, methods and procedures that place the user at the centre of an iterative design process (Bordin and Angeli, 2016). UCD adds UX techniques to the development process to support the design of a better experience for the user (Salah et al., 2014).

Several systematic literature reviews (SRLs) have addressed the integration of UCD, UX design and agile methods. These focus on aspects such as: (i) practices, techniques, evidence and recommendations (Wale-Kolade et al., 2013; Brown et al., 2011; da Silva et al., 2011); (ii) technical and non-technical gaps and trends (Salah et al., 2014; Jurca et al., 2014; Brhel et al., 2015); (iii) artefacts to report and control requirements (Schon et al., 2017), and (iv) the role of artefacts in the communication of aspects of UX in agile practices (Garcia et al., 2017). Recently, da Silva et al. (2018) discussed the evolution and current state of agile and UX based on a literature review. In the early days of agile, agile and UX design were assumed to work alongside each other; however, nowadays there is a recognition that agile and UX can no longer be seen as separate areas and that a shared understanding is needed. For example Kashfi et al. (2017) reported challenges, such as the impact of UX models, tools and methods in industrial settings, the need for formats other than text to communicate UX-related requirements, and the need for traceability between UX-related and business requirements.

Other researchers have explored the integration of UX and agile through practitioners' views. Ferreira et al. (2012) suggested promoting integration by mutual awareness of agile and UX team members and their roles in the development process, and by improving the engagement of both during the development process. Kuusinen (2015) suggested a task-oriented integration approach if UX experts are scarce, clarifying which UX-related tasks could be allocated to a developer and which require UX expertise. The role of developers' mindset in the integration of agile with UX has also been highlighted. Ardito et al. (2014) found two main causes for UX neglect: the lack of suitable UX methods with low resource requirements; and, that developers focus mostly on coding activities. McInerney (2017) investigated how the views of three UX experts changed with experience in agile development, interviewing the same individuals over a period of 12 years. In the more recent interviews, the UX experts indicated a preference for agile instead of the waterfall approach, which was not the case in earlier interviews. When working in an agile way, they could quickly see designs being implemented and could collaborate closely with developers. The UX experts also reported that more recently agile developers had started to pay more attention to UX.

Bordin and Angeli (2016) found that the limited involvement of users during an agile project and the lack of UX documentation led to problems with communication. As a solution, the authors recommended the adoption of UX techniques that require less time and effort, and a clear assignment of responsibilities for UX activities and decisions. Cajander et al. (2013) also found that it was unclear who was responsible for UX in an agile team, and that evaluating with users was the most commonly-used UX method.

Communication between agile and UX experts is made difficult by the different perspectives of the two parties as indicated by Larusdottir et al. (2017); whereas in agile practices the focus is on communication within the team, for UX professionals the emphasis is on communication with users. As a consequence, the methods and artefacts used in agile development and UX design are different and with little integration of information. The authors suggested that agile teams should have a clear vision for UX in an early development phase and refer back to it regularly to check whether it should be changed. Some authors suggested that an artefact-mediated communication approach might reduce communication problems in the conversation between practitioners from different areas (Brhel et al., 2015; Kashfi et al., 2017; Schon et al., 2017; Garcia et al., 2017). According to these authors, artefacts support the introduction of a common language to avoid misunderstandings and misinterpretations. Garcia et al. (2019) investigated an online agile community to understand how artefacts are used to mediate communication between UX and agile professionals. The results revealed that mock-ups, sketches of user interfaces and personas are the most used UX artefacts that support conversation in agile teams. The findings also pointed out that these artefacts are adopted in combination with user stories.

A diversity of virtual and physical environments have arisen to support teamwork and these have impact how information flows (Brown et al., 2011; Garcia et al., 2017). Virtual environments such as JIRA[1] and TFS (Team Foundation Server)[2] have been widely employed in global software development in an effort to reduce the distances in communication and collaboration amongst agile team members distributed geographically (Lanubile et al., 2013; Deshpande et al., 2016). Co-located teams also use these tools to manage the diversity and amount of information created, transformed and flowing daily in their work (Akman et al., 2016; Matta and Marchesi, 2015; Brown et al., 2011). Physical artefacts and tools, such as the Kanban board, continue to be part of the agile workplace (Deshpande et al., 2016; Brown et al., 2011; Liskin et al., 2014) and are used frequently for face-to-face communication, in stand-ups and planning meetings (Garcia et al., 2017).

Our work is motivated by existing research but differs in several ways: we emphasise the kind of UX information that an agile team handles, going beyond just identifying which UX artefacts are used; by focusing on the flow of information in the work of an agile team, we also look into how artefacts are used, how they support communication and awareness of UX, and how developers are involved in the creation and transformation of UX information, an activity known to enhance the shared understanding within teams.

[1] https://www.atlassian.com/software/jira

[2] https://docs.microsoft.com/en-us/tfs/

2.2. Theoretical underpinnings

We applied three complementary lenses in our study, described in more detail below. *Distributed Cognition for Teamwork* (DiCoT, Furniss and Blandford, 2006) was used to explore communication and collaboration around UX in the team's daily work. As the focus of our study was a small team, DiCOT provided an ideal framework for applying Distributed Cognition principles (Hutchins, 1995) in a practical setting. To help distinguish and categorise UX information, we drew on two well-known models of UX which provide complementary perspectives: Garrett's (2010) set of UX elements and planes, and Hassenzahl's (2010, 2018) content-oriented model.

2.2.1. Distributed Cognition for Teamwork (DiCoT)

Distributed cognition (DCog) considers that cognition manifests itself at the systemic level, rather than at the individual cognitive level of a person (Hutchins, 1995). According to DCog, information acquisition and propagation occur through the interaction of people, their environment, and artefacts. And that all of them affect human work (Hollan et al., 2000). DCog is a powerful framework that has been applied in airline cockpits (Hutchins, 1995), software programming (Mangalaraj et al., 2014) and in requirements engineering (Buchan, 2014). It has also been adopted by researchers in HCI (e.g. Wright et al., 2000) and CSCW (e.g Hoadley and Kilner, 2005) to analyse collaborative work and identify breakdowns in communication. In particular, informal communication can breakdown when the parties involved don't have a common memory of the conversation and what was decided.

Applying DCog can be challenging, and the DiCoT method (Furniss and Blandford, 2006) was developed to support the application of DCog principles when studying small teams. It combines the ideas and models of contextual inquiry (Beyer and Holtzblatt, 1998) with a set of 22 DCog principles (Furniss and Blandford, 2006) and provides a set of guidelines that the researcher can follow. The full DiCoT method categorises these 22 principles into five themes, three of which have been found to be useful in studying collaboration and communication in distributed (Sharp et al., 2012; Deshpande et al., 2016) and co-located (Sharp and Robinson, 2008) agile teams.

These themes are listed below and the related principles are shown in Table 1 (Furniss and Blandford 2006):

- artefact: concentrates on the structure of tools and representations, and how they affect work in practice;
- physical: focuses on the layout of the workplace and how it impacts the propagation of information;

- information flow: focuses on the communication between team members and how information flows and is transformed within a work setting;

Table 1: DiCoT principles - adapted from Furniss and Blandford (2006)

Artefact	
Mediating artefacts	used to support activities
Creating scaffolding	how people use the environment to support their tasks
Representation-goal parity	how artefacts in the environment represent the relationship between the current state and goal state
Coordination of resources	resources can be internally and externally coordinated to aid action and cognition (e.g. plans, goals, history, and so on)
Physical layout	
Space and cognition	the role of physical layout to support cognition
Perception	how spatial representations provide support for cognition
Naturalness	how closely the representation matches the properties of what it represents.
Subtle bodily supports	any bodily actions used to support activities.
Situation awareness	how the team are kept informed about the work through what they can see, hear and is made accessible to them
Horizon of observation	what people can see or hear (influences people situation awareness)
Arrangement of equipment	How the physical layout of equipment affects the access of information
Information flow	
Information movement	the mechanisms (representations and physical realisation) used to move information around the cognitive system
Information transformation	why, how and when information is transformed as it flows through the system
Information hubs	central points where information flows meet and decisions are made.
Buffers	where information is held until it can be processed
Communication bandwidth	the richness of different communication channels, e.g. face-to-face communication, computer-mediated communication, and so on
Informal and formal communication	the formality of communication, e.g. ad hoc conversation or planned meeting
Behavioural trigger factors	cause activity to happen without an overall plan needing to be in place

2.2.2. Models of UX

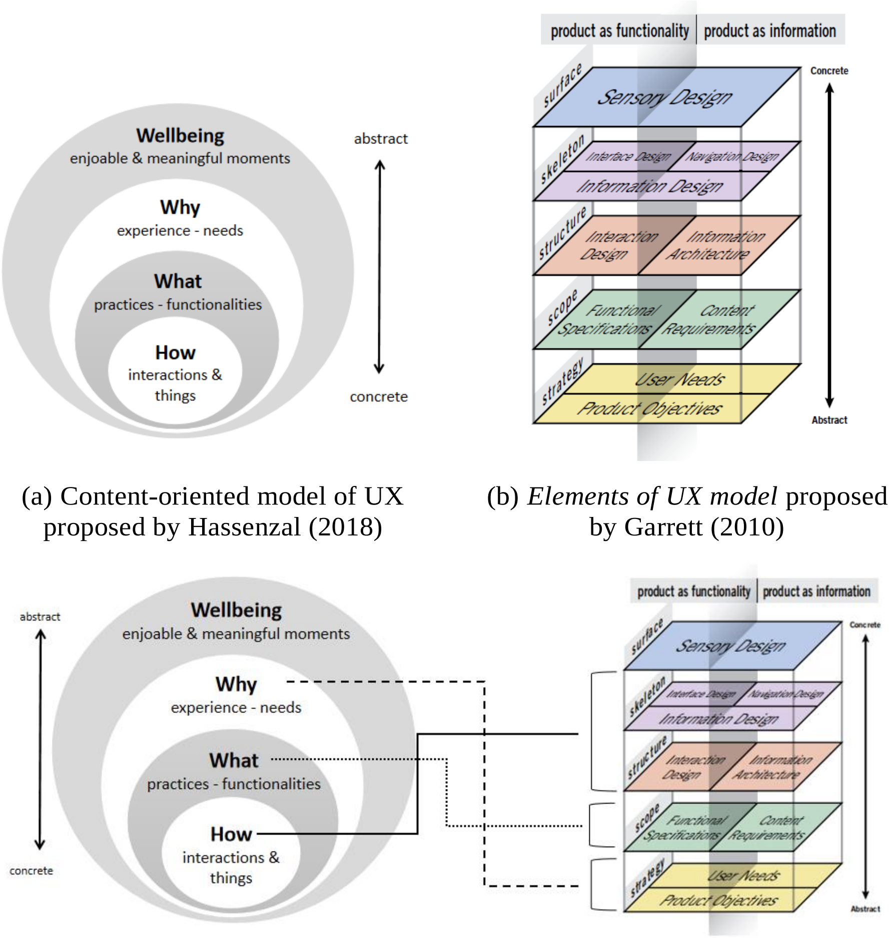
There are many models that describe the UX design process, but in our work we were aiming to identify the type of UX information considered by an agile team, rather than the processes undertaken. Few models conceptualise UX information or UX itself and we couldn’t find a single model that would be satisfactory for this purpose; instead, we identified two that provide complementary perspectives.

In 2010, Hassenzahl introduced a process-oriented model to illustrate how the design of the user experience is built from a designer perspective and from a user perspective. He also discusses that users are motivated by goals they want to reach which determine the interactions required to fulfil those goals. Considering this, he also introduced initial ideas of a three-level hierarchy of goals (Hassenzahl, 2010).

In 2018, he evolved this hierarchy of goals to propose a second model, a content-oriented model of UX (Hassenzahl, 2018) that focuses on the user perspective. This model contains three levels (*Why*, *What* and *How*) and a goal, the *Wellbeing* of users (Figure 1a). Each level guides the design of experiences that are mediated by a product. The *Why* level, the most abstract one, discusses the motivations that lead users to interact with a product; it focuses on the meaning that this interaction brings for the users. The *What* level is about the functionalities a product offers to users to fulfil their needs. The *How* level addresses concrete actions of the users for interacting with the tangible product, e.g. the arrangement of user interface elements. In this work, we adopted the content-oriented model of UX rather than the earlier process-oriented model because our focus is on UX information rather than UX processes.

Garrett (2010) proposes a different model of UX, the *"elements of UX"* framework (Figure 1b); this model was originally developed for web interfaces. It comprises five horizontal UX levels (called planes) with the more abstract plane at the bottom and the more concrete plane at the top. Elements of one plane may influence adjacent planes. The *strategy plane*, at the bottom, aligns users' needs with the product's objective. Next plane up, the *scope plane*, gives the detailed functionality of the product and the content elements required to meet the users' needs. Then, the *structure plane* specifies how the system interacts with a user and the arrangement of content. The *skeleton plane* deals with concrete elements such as buttons, fields and menus (interface design), content representation (information design) and with the interaction through the user interface (navigation design). Finally, the aesthetic elements are considered, in the *surface plane*, for a pleasing interface and fulfilment of the goals of the other planes. The five planes are split vertically into two perspectives: the *product as functionality* side reflecting the users' interaction with the product to accomplish tasks; and the *product as information* side focusing on what information the product offers to users to accomplish their tasks.

The two models (Figure 1c), complement each other and together provide a suitable framework for our analysis. Hassenzahl's model (2018) allowed us to take an abstract view in which UX is seen as the experience that a product can provide for the user and hence emphasises the user perspective; Garrett's model supported us with a concrete viewpoint to observe UX from the perspective of product construction. Both models emphasise that answering the *Why* first is crucial to be able to determine the features (i.e. the *What*) and the forms of interaction (i.e. the *How*); however, they provide different lenses. In Hassenzahl's model, the *Why* level focuses on understanding the effects that the product usage could have on the users' *wellbeing*; in Garrett's model, the *strategy plane* focuses on collecting product requirements from the end-users' perspective. The *What* level (Hassenzahl's model) and *scope plane* (Garrett's model) both have the purpose of defining the functionality of the product, but Garrett focuses on the functional specification of features while Hassenzahl takes an integrated view of the users' overall experience. Finally, the *How* level (Hassenzahl's model) considers the users' actions with the product, and Garrett's three top planes (i.e. *structure*, *skeleton* and *surface*) specify the mechanisms to support the users' actions. Garrett's model offers a more granular description of the elements that directly impact the design of interaction.



(a) Content-oriented model of UX proposed by Hassenzal (2018)

(b) *Elements of UX model* proposed by Garrett (2010)

(c) The complementary view of Hassenzahl and Garrett models

Figure 1: Models of UX adopted in this work

3. Research method

Workplace studies aim to understand not only the professionals' activities but also the rationale underlying their actions in daily work (Ferreira et al., 2012). These studies take into account the social and situated aspects of work providing a holistic view of the whole workplace (Sellberg and Lindblom, 2014). Ethnography is one research method commonly used with DCog in workplace studies. It is a qualitative empirical approach that aims to provide an understanding of people and cultures situated in their social environments. In the case of software engineering studies, the social environment is recognised as the professionals' workplace. An ethnographic study differs from other qualitative research methods in that it stimulates the researcher to have a strong engagement with the participant group (Sharp et al., 2016).

Our study consisted of four main parts: (1) pre-planning, (2) study planning, (3) data collection, and (4) analysis (Figure 2). Many iterations were performed to gather data in the organisational setting by repeating steps (2) to (4). Data collection and analysis were guided by the DiCoT principles discussed in Section 2.2.1. The second author established contact with the organisation. The study design, data gathering and analysis were mainly performed by the first author. The second and third authors participated in some observation sessions and meetings, and both supported the reviewing and refining of the study steps. The data analysis was intensely discussed by the three authors.

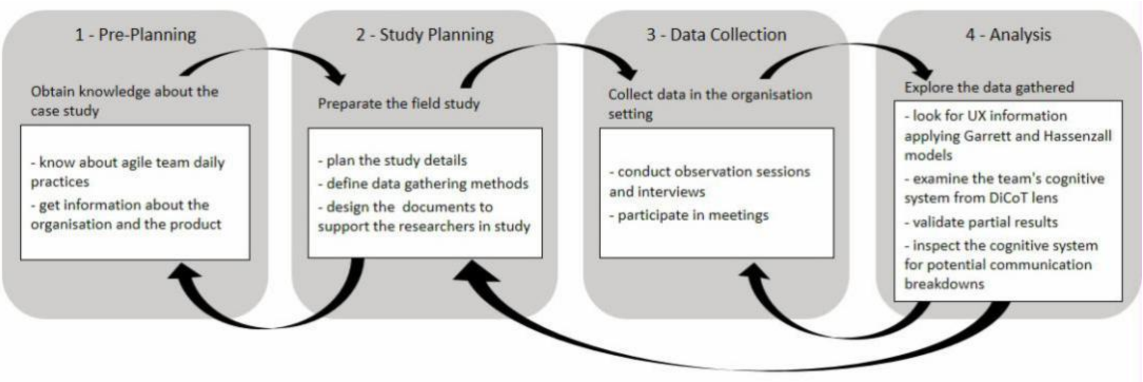

1 - Pre-Planning
Obtain knowledge about the case study
- know about agile team daily practices
- get information about the organisation and the product
2 - Study Planning
Preparate the field study
- plan the study details
- define data gathering methods
- design the documents to support the researchers in study
3 - Data Collection
Collect data in the organisation setting
- conduct observation sessions and interviews
- participate in meetings
4 - Analysis
Explore the data gathered
- look for UX information applying Garrett and Hassenzall models
- examine the team's cognitive system from DiCoT lens
- validate partial results
- inspect the cognitive system for potential communication breakdowns

Figure 2: Overview of the research method

3.1. Study context

The study was carried out in the software development unit of a large university's IT function based in the United Kingdom (UK), with more than 100 developers working on building products to support different areas of the university including management information, student support and course design. The university's VLE (Virtual Learning Environment), a version of Moodle (docs.moodle.org), was the project chosen for our study. The VLE platform provides resources for running courses at a distance. The different modules of the VLE are used by many thousands of users (educators, tutors, students).

Five co-located agile teams worked on the VLE project which was sliced into sub-projects. The teams used the Scrum framework to support their agile practices (Rubin, 2012). Each team had around five developers, one of whom played the additional role of team leader, and one Scrum Master (SM). Developers adopted cross-functional roles acting as developers and testers. Nine Product Owners (POs) participated in different VLE sub-projects, typically two POs in each sub-project; they were experts in different VLE modules (e.g. learning tests, students' statistics, delivery of materials) and represented the business view. Business analysts, user interface (UI) designers (who focused on screen design) and UX designers (who carried out research and evaluation with end-users), members of the helpdesk team, and project manager were other professionals that interacted with the VLE project. The developers and SM were co-located; the POs were co-located in another building separate from the developers. Development followed 2-week sprints, or iterations, and the teams used Scrum ceremonies e.g. release planning meetings, sprint meetings, and stand-up meetings.

3.2. Study planning

Planning started with two meetings of the two first authors with an SM. In the first meeting, the researchers gained an overview of how teams worked and what the project was about, by asking two questions: (i) *how does a typical VLE sub-project run?* and *(ii) what are the agile practices they use?*. In addition, documentation about the VLE project was explored. In the second meeting, the SM and the researchers agreed on the boundaries of the study. Permission was obtained to conduct contextual interviews and observation sessions with one VLE team. Documents were explored only with the authorisation of team members. The SM suggested observing an agile team whose members were experienced in agile.

After these meetings, the study was formalised using the five ethnographic dimensions as proposed by Sharp et al. (2016):

(i) *observation type* (participant or not) was non-participant observation with the researcher asking questions and observing individuals performing their tasks;

(ii) *duration of field study* was estimated at 3 months;

(iii) space and location where the observation will happen was the software development unit;

(iv) *theoretical underpinnings that support the study* consisted of DiCoT (Furniss and Blandford, 2006), Garrett's framework (Garrett, 2010) and Hassenzahl's content-oriented model of UX (Hassenzahl, 2018);

(v) the ethnographers' intent in undertaking the study was defined by the RQs (see Section 1).

A paper form with the DiCoT themes and principles was designed and used in all observation sessions.

3.3. Data collection

The investigation was conducted over 20 days between 22nd March and 17th May 2018. Interviews were carried out to obtain further details and clarifications after a set of observation sessions. Semi-structured interviews were constructed based on analysis of the data collected from observations. The first interview was with the SM to get an overview of teamwork. Some interviews took place in June 2018 to validate whether the interpretation of the data collected was correct.

Each observation session lasted between 1 and 3 hours, and sessions were conducted on two non-consecutive days each week. The researchers and the SM agreed on the best days respecting the demands of the team. The observations comprised: team member individual work, stand-up meetings, one release meeting, and one demo meeting. The data was collected through different media: audio recordings, handwritten notes, photographs and digital copies of documents and artefacts. To complement the data gathered, the first author drew sketches of the workplace layouts. DiCoT models guided all data collection.

The kick-off question in the first interaction with a team member was: *How does your daily work run? Please explain, giving examples of artefacts, task and ways of communication that you and other team members use*. When asked about *the UX information flow within the agile projects*, all interviewees insisted that they were not responsible for giving information or preparing artefacts about UX, so we decided to ask more about *"the UX artefacts they used and who was responsible for their creation"*. We deployed a questionnaire to collect demographic data of team members. Table 2 summarises the profiles.

Table 2: Demographic data of the team members

Participant	**Experience in that role at OU**	**Experience In agile development**	**Roles played in the team**	**Expertise in VLE domain**
#1	18 years	> 10 years	developer, tester, leader	Expert
#2	12 years	4 years	developer, leader, delivery manager	Proficient
#3	11 years	2.5 years	developer,tester	competent
#4	10 years	1.5 year	developer	competent
#5	3 years	5 years	developer	expert
#6	2 years	6 years	scrum master	expert

The first author also participated in a workshop in which all the agile teams working on the VLE project (including POs and other collaborators) discussed the process they were currently using for product development and delivery. UX was not discussed explicitly, however participants expressed concerns about not using the feedback collected from end-users as well as they could. From our perspective, the workshop was key to understanding how information from other collaborators entered the cognitive system. Table 3 summarises the study instruments.

3.4. Analysis

Analysis was conducted in an iterative process in which the three authors discussed the data collected using the three DiCoT themes. The first author arranged the data weekly after each data collection session (i.e. observations sessions, interviews). Then, a first analysis of the data was conducted to get some insights or make adjustments to the planning for subsequent collection activities. Additionally, any uncertainties about the data collected were clarified soon after with the team.

The NViVo tool[3] supported the first author in managing the data. The three authors met regularly to discuss the results. Successive iterations of writing, discussing and reflecting helped to shape the findings.

Table 3: Study instruments and data gathered

Method applied	**Source and media**	**Data collected**
Interviews (team members)	8 hours of audio, 13 pages of handwritten notes and 10 photographs.	overview of daily and members perspective of their own work.
Interviews (member checking)	2 hours of audio, 3 sketches were drawn.	validation of the researchers' interpretation of data; workplace features.
Observation sessions (daily work)	6 hours of observation, 10 pages of handwritten notes, 3 hours of audio and 3 photographs.	members interaction during meetings (stand-up, release and demo) and daily work.
Observation session (workshop)	4 hours of observation, 20 pages of handwritten notes and 4 photographs.	interaction and discussion of 30 participants during the workshop.
Artefacts exploration	4 documents and 12 artefacts.	characteristics of the artefacts and their relationship with team work.

The iterative analysis process was organised in two rounds: in the first round, the authors looked for explicit evidence of UX in all the data, e.g. use of a UX artefact. When evidence of UX was identified, tags were assigned to that chunk of data and notes were added to it. During this round, we also attributed codes to identify the ceremony in which the evidence arose or the artefact was used. During this step, the relationship among artefacts, people and activities was coded. We also took extracts from audio recordings of the interviews that could explain some of our findings.

[3] https://www.qsrinternational.com/nvivo/home

In the second round, the set of data was analysed using the DiCoT themes (see Table 1). This led us to an understanding of how UX information featured in the cognitive system and its propagation. Finally, we inspected the analysis for potential communication breakdowns in UX information flow.

4. Findings

We first present the findings according to the DiCoT themes: artefact (4.1), physical (4.2) and information flow (4.3). This analysis focuses on identifying the artefacts that contain UX information, how they are used, and how UX information flows within the agile team's daily work, in order to determine where UX features within the team's cognitive system, answering RQ2. We then apply Hassenzahl's and Garrett's lenses to investigate the type of UX information within this system (4.4) answering RQ1. Some aspects of the findings reflect more general issues raised by DiCoT analyses of agile teams, e.g. the importance of stand-up meetings as key co-ordination activities (Sharp and Robinson, 2006), but we aim to emphasise findings related specifically to UX information and only provide more general findings for completeness.

4.1 Artefacts

Furniss and Blandford (2006) state that there may be many artefacts within a cognitive system that support the coordination of activity and recommend that analysis should concentrate only on those that are relevant to the study. In our case then, we focus only on those artefacts that impinge directly on daily work of the agile team, and that relate to UX. These were identified through early cycles of data collection including interviews with team members and initial observations. Consequently, we analyse six artefacts that carried UX information or impacted upon it. These are: user stories (represented as story cards or tasks), the Kanban board, a virtual tool to manage team work (Team Foundation Server), the handover checklist, mock-ups, and the newsboard. The Kanban board, story cards and newsboard were physical artefacts while the others were virtual. The Kanban board and Team Foundation Server (TFS)[4] were key *mediating artefacts* that supported coordination of team work. Both of them can be seen as containers, i.e. they hold other artefacts together in one place (Liskin, 2015). Developers, code reviewers, testers and POs tracked their goals and coordinated their activities by following information in TFS and the Kanban board. We describe these artefacts in detail below, drawing on the DiCoT principles to describe their role in the cognitive system.

[4] TFS is a Microsoft product that covers the entire product lifecycle, including source code, requirement and project management.

4.1.1 User stories

User stories were captured in two places: on the Kanban board as story cards, and in TFS as tasks. Story cards (sticky notes) were displayed on the Kanban board. Each story card represented a user story which had been created in TFS previously. Each card contained a hand-written user story name and an identification number corresponding to the task in TFS, for use during daily stand-up meetings. The colour of a sticky note represented which sub-project the story was related to, and coloured dots stuck on the notes denoted additional information, such as a yellow dot indicating that more information about that story was needed before going ahead. A template to guide story card elaboration was provided. No other information was captured on the story card.

User stories were also captured as tasks in TFS. Two formats were used for describing the user stories. The first followed the common user story template "As a <type of user> I want <some goal> because <some reason>" (Cohn, 2009). In the second, user stories were written in a free format describing, for example, step by step how a feature works and its technical details. Team members reported that they did not follow a pattern as to which format was used when.

4.1.2 Kanban board

The Kanban board supported the *coordination of resources* and was used to represent sprints (Figure 3). It contained story cards in the form of sticky notes in different colours, which were placed in one of five columns to show which implementation stage the story had reached; these stages were: committed to sprint; in progress; ready for review; in test; or that the test was finished and the implementation was ready for verification by the PO. During stand-up meetings, the story cards on the Kanban board were moved between the columns and together they kept a record of progress on a daily basis.

Representation-goal parity was found in the positioning of story cards in different columns on the Kanban board to represent the overall progress of activities. As is commonly the case, the structure of the Board was designed to emphasise that the stories need to move from the left-hand side of the Board to the right-hand side, i.e. to the last column for PO approval.

The information on the Kanban board complemented that available in TFS' backlog (see below).

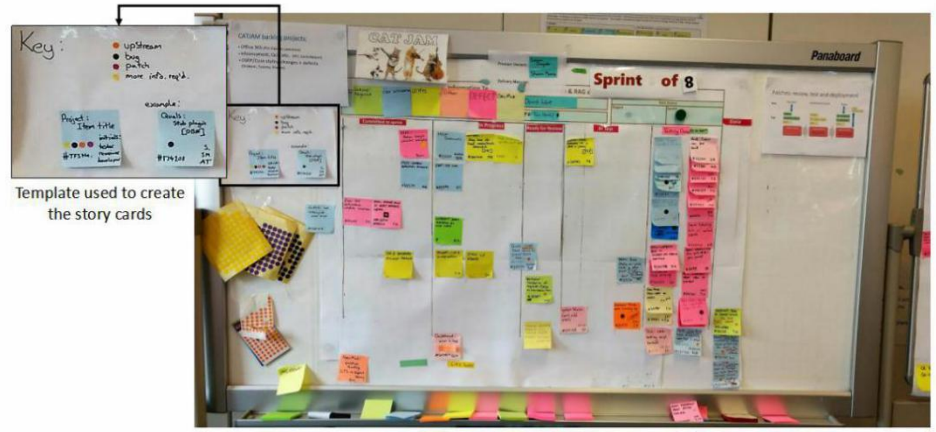


Figure 3: Physical Kanban board showing coloured story cards and annotations using coloured dots (colours are orange (upstream); green/black (bug); red (patch); yellow (more info required)

4.1.3 Newsboard

The newsboard was a wall-mounted physical board placed at the entrance to the team's room. Feedback from users about the VLE platform, collected after product release, was displayed on the board. Printed copies of user interface screens were fixed on the board, and speech bubbles representing user's voices contained user feedback. Figure 4 shows a diagrammatic representation of the newsboard.

[Participant #6] reported *"…UX designers and POs report the users feedback during the retrospective meetings. However, there are no formal representations of users' feedback (model, etc…"*. Yet during the workshop the participants mentioned that although the IT department frequently conducted research to gather users' feedback, the developers had access to few results; they suggested that the results of user research should be permanently available to the software development unit. None of them cited the newsboard as a source of UX information, indicating that this artefact was not seen as relevant to their work.

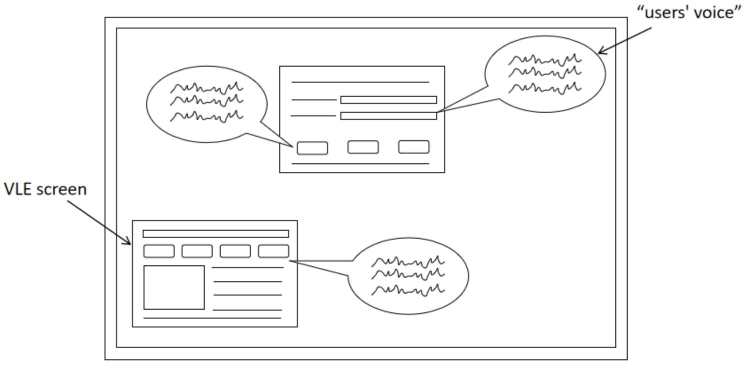
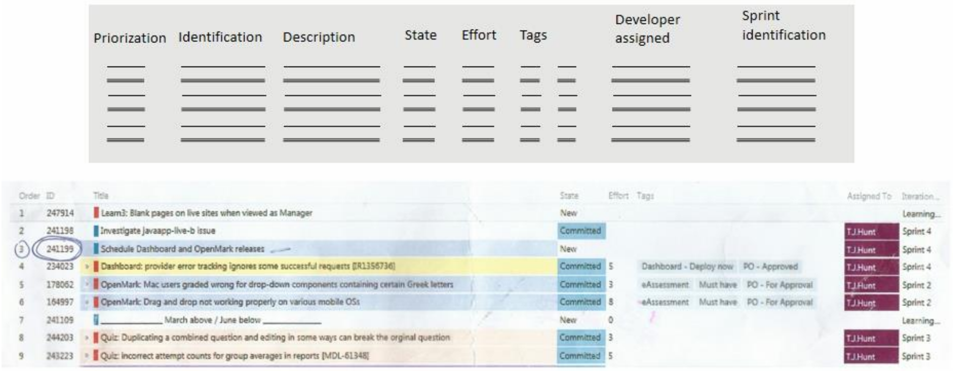


Figure 4: Diagrammatic representation of the newsboard

4.1.4 Team Foundation Server (TFS)

TFS supported the *coordination of resources* and held all the project information in one place. It provided different ways to visualize this information, with the backlog and the task views being the most used by the team we observed.

The backlog view showed an overview of all the tasks (user stories). Figure 5 contains a diagrammatic representation of the backlog and example photograph. From the backlog, the team could see the main task details: its priorization, an identification number, brief description, where it is in the development process, the estimated effort to do it, the assigned developer, and the sprint to which the task belonged. In addition, tags could be added to provide additional information like that represented on the Kanban board by coloured dots, such as whether the team had requested more information from the PO.



Order	ID	Title	State	Effort	Tags	Assigned To	Iteration...
1	247914	Learn3: Blank pages on live sites when viewed as Manager	New				Learning...
2	241198	Investigate javaapp-live-b issue	Committed			T.J.Hunt	Sprint 4
3	241199	Schedule Dashboard and OpenMark releases	New			T.J.Hunt	Sprint 4
4	234023	Dashboard: provider error tracking ignores some successful requests [IR1356736]	Committed	5	Dashboard - Deploy now PO - Approved	T.J.Hunt	Sprint 4
5	178062	OpenMark: Mac users graded wrong for drop-down components containing certain Greek letters	Committed	3	eAssessment Must have PO - For Approval	T.J.Hunt	Sprint 2
6	164997	OpenMark: Drag and drop not working properly on various mobile OSs	Committed	8	eAssessment Must have PO - For Approval	T.J.Hunt	Sprint 2
7	241109	____________ March above / June below ____________	New	0			Learning...
8	244203	Quiz: Duplicating a combined question and editing in some ways can break the orginal question	Committed	3		T.J.Hunt	Sprint 3
9	243223	Quiz: incorrect attempt counts for group averages in reports [MDL-61348]	Committed	5		T.J.Hunt	Sprint 3

Figure 5: Diagrammatic representation and a photograph of the backlog view format in TFS

The task view contained many details of the project, the team, the task and acceptance criteria. Additionally, complementary information could be added to any field through external links to repositories and tools, or as attachments, e.g. a mockup (see below). Figure 6 shows a diagrammatic representation of the task view.

As a dynamic artefact, TFS *created scaffolding* for the team in the form of notifications that were triggered when tasks were updated, bringing any changes to the attention of team members.

TFS contained many other artefacts, and had a complex structure, which became particularly evident for our focus on UX information. This was illustrated by the difficulties we observed when team members were trying to find information that required a number of artefacts to be used together. For instance, when we asked participant #1 to explain a feature he was handling, he opened a number of different screens in TFS and followed several navigation paths to find the information.

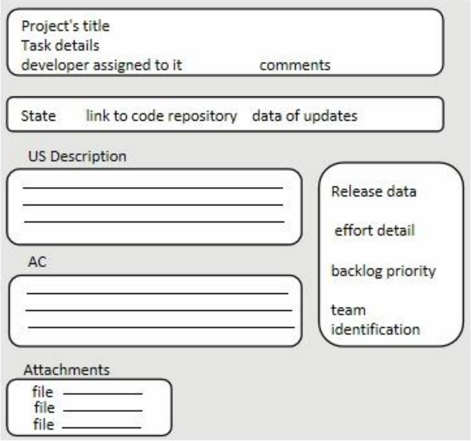


Figure 6: Diagrammatic representation of the task view in TFS, as used by this team (US = user story; AC = acceptance criteria)

In the interview [participant #6] stated that *"TFS is the main place where information is shared"*. However, difficulties relating to making information visible and accessible from TFS were also mentioned. [Participant #2], for instance, mentioned *"...the backlog of TFS does not provide enough resources to represent all the information we need... so we use different elements on the Kanban board to represent our work..."*. [Participant #1] said *"...we are not able to set up in TFS our process of testing and reviewing of coding... to do this we need to create some extra tasks..."*. These difficulties were also evident in our observations, and were mentioned explicitly during the process workshop, when some participants pointed out that TFS was not flexible enough to accommodate the team's development practices. In TFS, the *representation-goal parity* of artefacts and the *coordination of resources* were neither explicit nor visible in the cognitive system.

4.1.5 Handover checklist

This *created scaffolding* for the team by reminding team members of the important actions they must do for each story before it can be evaluated by the PO. It contained a set of questions with Yes/No options, sorted by topics related to quality requirements and documentation updates. All team members used the same handover checklist.

The handover checklist was embedded as an attachment in TFS and was linked to other official project documents such as web standards, accessibility and usability checklists. The checklist questions were completed as the story progressed through the different stages of development, e.g. from coding to testing, helping with the *coordination of resources*. After completing the checklist, the story state changed to signal it was ready for PO evaluation.

All team members cited the handover checklist when questioned about artefacts which supported their work. For example, [participant #5]: *"...we run [handover] by ourselves before we send code for review so this checks basically common things which we might have missed... things which we need to take into account as accessibility checks..."*; and [participant #6]: *"...handover plays the role of a reminder of what the developers should not forget to do before deploying the code into Git"*.

4.1.6 Mock-ups

Mock-ups are a visual representation of the user interface associated with a user story. Developers followed these when coding features. Usually, they were attached to the task as a PDF document in TFS. Mock-ups *created scaffolding* for the team. The mock-ups gave details of user interface elements, formats, styles, positions on the screen, label names, and so on. They also had *representation-goal parity* as the intended interface (the goal state) was captured in the mock-up.

Team members reported that UX and UI designers were both responsible for the creation of mock-ups with the support of a PO: [participant #5]: *"I discuss technical issues about UI development generally with another developer... about what it should do, what it should look like, the language we should use...in some cases if there is a lot of new user interface work as part of the project, we will have a UX designer working with the PO..."*; [participant #2]: *"UI designers are not part of the team...The mock-up shows the position that an element should be ...it gives specific marks on the screen, colours, style guides..."*.

Sometimes, developers made sketches of the user interface (Figure 7) to elaborate on the mock-ups, when meeting one-to-one with their PO, or to capture information from a meeting. These sketches *created* temporary *scaffolding*, and were drawn by developers to confirm their understanding of what was required and then used during the coding activity.

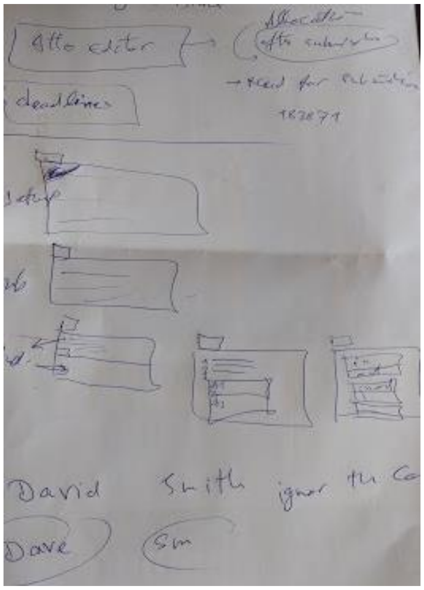


Figure 7: Example of sketches drawn by a developer during a one-to-one demo meeting with the PO. These formed temporary scaffolding

4.2 Physical (and virtual) layout

Through our data collection it soon became clear that virtual layout of information was as important to the agile team as the physical layout; for example, the team's *situation awareness* relied on virtual artefacts such as mock-ups and handover checklists that were updated regularly. Hence, we consider below both physical and virtual layout, starting with physical layout.

The software development unit was located in a large room and team members were mostly spread out. Figure 8 shows the room layout and its dimensions; the circles identify where members of the team we observed were sitting; other workspaces were occupied by members of other teams. The Kanban and newsboard were located at one end of the room.

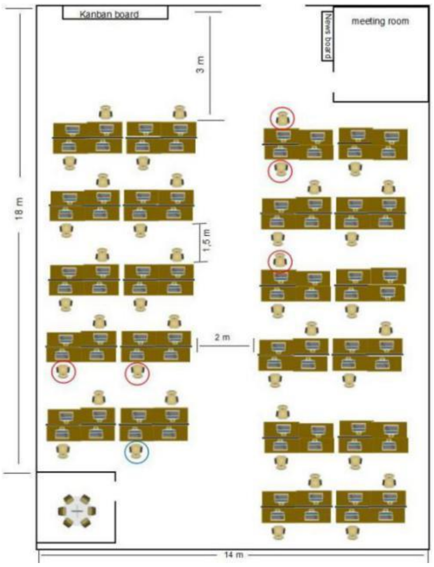


Figure 8: Physical room layout illustrating the seating plan for the team members in the office (developers - red circle and SM - blue circle)

Due to the large dimension of the room and the location of the Kanban board, developers could not see story cards easily, nor their detail. This *arrangement of equipment* affected their *horizon ofobservation* and *situation awareness* of ongoing activities. The newsboard was visible only when the members were exiting or entering the room. This artefact attempted a *naturalness* of representation by using speech bubbles to capture users' comments.

Release and sprint planning meetings took place in a nearby building where POs, UI and UX designers were located. They were supported by a large display screen which was controlled by a computer. Participants sat around a large table and shared the TFS backlog, mock-ups and simulations of features displayed on the screen, allowing everyone to see the virtual information during face-to-face discussions. Patterns of *subtle body supports* were identified specifically during the meetings such as movements in the air to draw some parts of a user interface or to simulate user interaction with the system.

TFS was visible to all team members throughout the day, via their workstations. TFS supported *situation awareness* virtually and provided a very long but hidden potential *horizon of observation*. The team's work was driven by user stories, but as the Kanban board was only reviewed once a day, TFS was key. However, UX information was embedded in TFS's tasks, and from a UX perspective, neither the principle of *space and cognition*, nor the principle of *perception* were applied effectively to support cognition. Figure 9 shows a diagrammatic representation of how UX information was accessed via deep navigation paths.

This illustrates the complexity around UX information: to gather it, team members had to follow virtual navigation paths within and outside TFS.

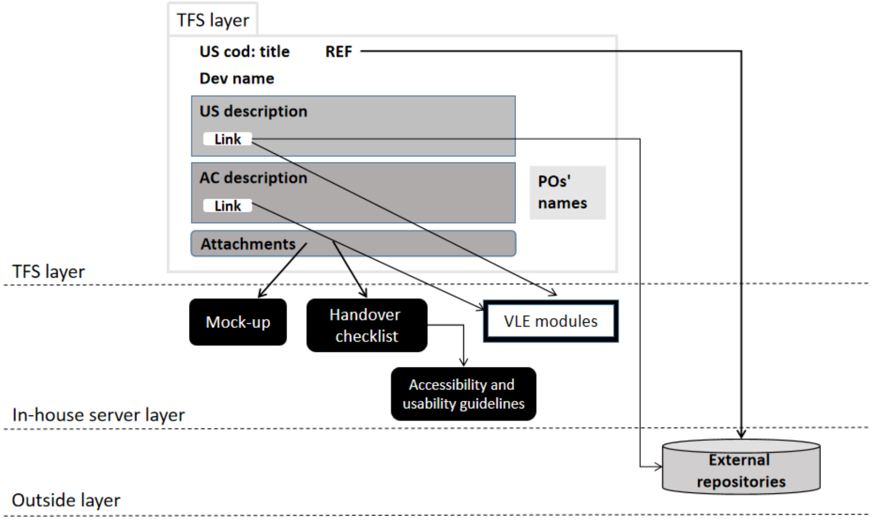


Figure 9: Diagrammatic representation of UX information embedded in TFS and its deep navigation paths

4.3 Information flow

Figure 10 presents an overview of how the information flows around this team's cognitive system. In creating this diagram, we focused on artefacts that the team used in their daily work, and any interactions where we could find evidence of UX information. We also included other professionals involved to illustrate the direct and indirect influence each one had in the information flow; developers (i.e. team members in Figure 10) received UX information via different pathways.

The inception of *information movement* into the cognitive system came from informal or formal requests from VLE users (i.e. educators, students, and so on) and sometimes UX designers. The requests reached the support team in the form of an electronic ticket input to the incident system. Additionally, the POs might receive *informal* or *formal* requests from educators and the technical support team. The POs or UX designers checked the viability and accuracy of the requests and added checked requests to the list of requests (*information transformation*).

The filtered requests were re-written by POs into tasks in TFS as a result of conversations between the POs and the agile team members in groups or individually (i.e. the information was *transformed*), and some UX information was added to some stories. The tasks may then be refined by developers, during sprint planning meetings. The most common information added to tasks at this point was the inclusion of details regarding acceptance criteria, technical requirements, or testing steps, i.e. a translation of the mock-up into a step-by-step interactive test sequence.

UX information was also *transformed* when the UI/UX designers constructed mock-ups for the tasks; these were added to TFS by themselves or POs. UI designers obtained insights and information from the POs and from the description found in the list of requests. If a project involved the development of new solutions, a careful study with users was performed by UX designers before the release planning meeting. [Participants #2 and #6] commented about the *transformation* of information from UX designers to mock-up development: *"...this data can come from PO's previous knowledge or from some more elaborate user research... In big projects the mock-ups are built considering outcomes of user research conducted previously"*. Depending on the level of detail, the PO would link some parts of the mock-ups by explicitly mentioning them in the task description. The handover checklist was added as an attachment for each task in TFS and developers, reviewers and testers assigned to a task completed the checklist through the implementation process.

The mock-ups and handover checklist were static artefacts from the agile team's point of view in that the team did not elaborate the UX information they contained and only received and applied the information provided to them.

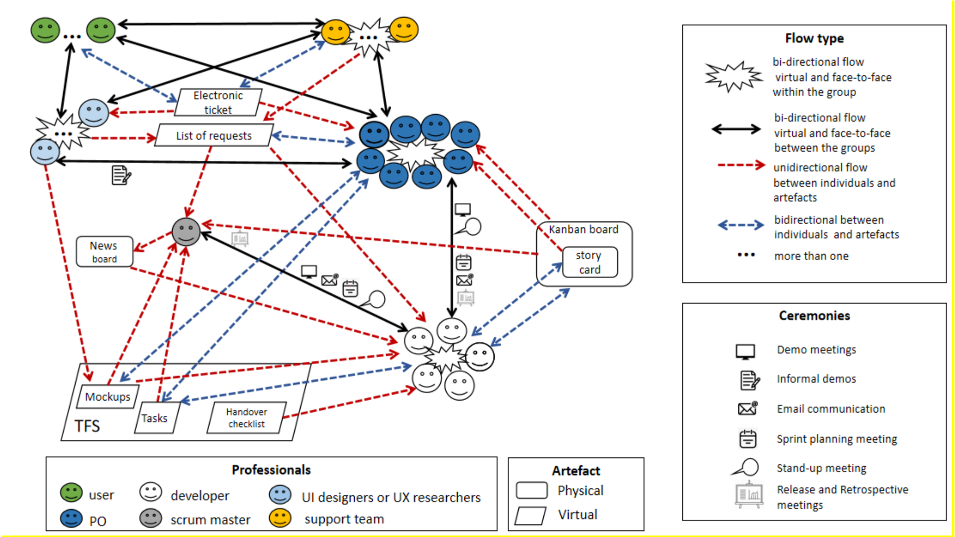


Figure 10: UX information flow within the team's cognitive system

Formal or *informal communication* about UX could arise in stand-ups, demos, release, sprint or retrospective meetings, although the focus was not always on UX. Examples of where UX information was exchanged and decisions made include user interface issues in the discussion of a story during stand-ups. During a demo or sprint planning meeting team members might recount typical situations of end-users in an informal way and suggest a context to illustrate possible solutions. For instance, [participant #2] mentioned *"...in the real life day, the user submits their work on the deadline..."*. During demos, team members used mock-ups and a user testing tool to demonstrate the status of the development by simulating how a user interacted with UI elements and information. In addition, team members had parallel conversations, *informal communication*, which commonly happened after stand-up meetings. For example [Participant #5] reported *"...Some ad-hoc demos can also be done at any time... there are particular issues that arise from development – usually raised at stand-ups."*.

Key decisions were made in stand-ups, demos, release, sprint and retrospective meetings indicating that these represent *information hubs* of the cognitive system. TFS and the Kanban board held information on which key decisions could be made, and hence were *information buffers*. However, we found no evidence of *information hubs* and *buffers* dedicated to UX in the agile team's cognitive system. Usually, all the decisions regarding mock-ups, accessibility and usability checklists were made by POs and UI/UX designers.

We did not identify any *behavioural trigger factors* specifically regarding UX information. Team members received UX information from POs or UX professionals as part of the standard development process. Although they may contact the POs or designers with queries, most of the time, the team was the receiver of UX information.

4.4 UX information within the team's cognitive system

The DiCoT analysis shows that UX information appears in some development artefacts and is discussed during some meetings and interactions involving agile team members, the POs and UI/UX designers. This section applies the UX models introduced in Section 2.2.2 and uses them to explore the type of UX information that teams handle in their daily work. We use the levels and planes of the UX models introduced in Figure 1 to describe the information we found.

4.4.1 UX information: Why

Traces of Hassenzal's *Why* level of UX information appear in face-to-face meetings and informal conversations, but only occasionally. For example, when discussing scenarios of use in demo meetings, developers and POs may talk about the goal of a feature, i.e. whether this feature meets the *user needs* (Garrett's model); end-users' feedback and results of user research were presented to the team during retrospective meetings.

The only artefact that showed traces of this level of UX information was the newsboard. Here, the speech bubbles together with the user interface screens showed traces of all three levels: the *Why*, *What* and *How* types of UX information. In particular, the bubbles contained descriptions of users' reactions when they used a feature. These descriptions reflected the *Why*, i.e. their motivations to use that feature or their feelings when using that feature.

4.4.2 UX information: What

As would be expected, user stories contain information regarding the *What* of a system (its features) and hence defined its *scope*. However, their main focus was on technical and business information rather than explicitly considering the user experience more holistically, or understanding the users' perspective. For example, the Kanban Board and story cards held only identification and technical information related to system structure and process, and did not include any references or pointers to UX information. In TFS, the main body of tasks contained *functional specifications* and *contents requirements*, however, they also linked to mock-ups and other artefacts that contained *how* UX information. The newsboard included some UX information at the *What* level from the users' point of view. For example, in the speech bubbles users are reacting a feature and hence are providing information in the *What* level.

We also see evidence of the *What* level in demo meetings and stand-ups – mostly in the form of discussions relating to features that are implemented through tasks.

4.4.3 UX information: How

In contrast to the other two levels, UX information that focuses on the *How* level appears in many places within the cognitive system. Specifically, this type of information was identified in mock-ups, newsboard, TFS tasks, and handover checklists. Mock-ups were concrete models of the VLE *user interface* and carried details regarding styles, *interface design* and *information architecture*, respectively *surface*, *skeleton*, and *structure planes* of Garrett's framework. The handover checklist guidelines, i.e. web standards, accessibility and usability checklists, contained aspects of *sensory design*, *interface design* and *navigation design*, belonging respectively to Garrett's *surface* and *skeleton* planes. All elements found in the mock-ups and guidelines were therefore related to the *How* level.

Most TFS tasks contained technical and business information, but some contained links to other VLE modules to give concrete examples of what the developer should implement, and others showed evidence of user interaction (*interaction design*), mandatory user interface styles (*sensory design*), and elements of the user interface (*interface design*). These also focused on the *How* level. There were no *structural* elements in tasks dedicated to UX. In some tasks there was evidence of the following UX elements: user interactions (*interaction design*), mandatory user interface styles (*sensory design*), and interface (*interface design*).

User stories and mock-ups took complementary perspectives on UX information when viewed through Garrett's vertical planes. Mock-ups provided the team with details of what users could see and which parts they could interact with, i.e. *product as information* in Garrett's framework. User stories, on the other hand, captured the product's behaviour by describing the user's interaction, i.e. *product as functionality* in Garrett's framework (see Table 4).

Table 4: "How" UX elements captured in user stories (tasks) and mock-ups within Garrett's vertical planes

Artefact	**Surface plane**	**Skeleton plane**	**Structure plane**	**Garrett's Perspective**
Tasks in TFS (User stories)	sensory	interface	interaction	product as functionality
Mock-ups	sensory	interface	architecture	product as information

The newsboard contained *How* level information too, e.g. the links between the bubbles and the *user interface* screens illustrated the concrete elements that made up the *user interaction* with the product.

We found evidence of *how* information in all the information hubs identified in the information flow model: stand-ups, demos, release, sprint and retrospective meetings. For example, in demo meetings, when the PO and developers were guided by a scenario of use, the conversation focused on *interaction design*, *information architecture* and *navigation design*; in sprint planning meetings, conversations focused on prototypes referred to *interface* widgets (e.g. buttons), potential usability problems, *navigation* (sequence of actions) and position of *interface* elements; and in stand-ups specific parts of the *interface* were discussed.

5. Discussion

In this section, we answer the research questions from Section 1, and extract recommendations for improving the flow of UX information in agile teams. To answer RQ1, we analysed the data using the models of UX described in Section 2.2.2 (Figure 1). DiCoT provided a structure to answer RQ1 and RQ2 by identifying where UX information could be found within artefacts, and how information flows and is transformed through the cognitive system. Throughout this section we also consider whether our findings may be applicable to other agile teams, by comparing our key insights with previous studies reported in the literature.

5.1 RQ1 - Which type of UX information do agile teams process in their daily work?

In our study, UX information appeared in artefacts and in interactions between members of the team, POs and UI/UX designers. Information relating to the *How* level, specifically the *surface*, *skeleton* and *structure* planes, were the most common type of UX information captured in the artefacts used in the daily work of the team. This type of information was identified in mock-ups, TFS tasks, newsboard and handover checklists. Information relating to the *What* level featured in user stories (specifically tasks in TFS) and also arose during verbal interactions between team members and POs or UX/UI designers. The newsboard contained traces of all three levels of UX information. All levels of UX information were mentioned to some degree in the various Scrum ceremonies and in informal interactions. The *Why* level was least well-represented in both artefacts and information flow, although was occasionally mentioned in meetings.

The handover checklist and newsboard artefacts are peculiar to our studied setting, but mock-ups and TFS tasks (user stories) are commonly used in agile teams. For example, several studies report that mock-ups are the artefact most often adopted by software professionals to represent UX work (Gray, 2016; Kashfi et al., 2017; Ardito et. al, 2014) and that mock-ups are used in combination with technical documents and user stories (Bordin and Angeli, 2016; Garcia et al., 2019). Others have also found that most UX information is discussed in agile ceremonies rather than captured in artefacts (Cajander et al., 2013; Garcia et al., 2019).

This comparison with other studies indicates that our findings regarding the type of UX information considered are relevant beyond the studied setting.

5.2 RQ2 - How does UX information flow within the cognitive system of an agile team?

In our study, UX information flowed within the cognitive system via the key information buffer TFS, and via synchronous communication in ceremonies and meetings, and asynchronous communication by email. However several aspects of this information flow are problematic.

First, some types of UX information, particularly the *Why*, flow almost exclusively through oral communication and are not captured in artefacts. This suggests an increased level of transformation of this type of UX information between people and representations, e.g. between individuals and through individuals' own notes or sketches. High levels of information transformation can be a positive sign of teams developing a shared understanding (Abdullah et. al., 2010), but it can also cause communication problems if understandings are not discussed sufficiently and aligned.

Second, UX information was gathered from users by POs, UI designers and UX designers, and transformed into user stories and mock-ups which were then delivered to developers. Accessibility and usability guidelines were also usually prepared by POs, UX designers and UI designers with no developer input. Developers therefore had no active participation in constructing UX information but only transformed it to produce code, supported by style guides and accessibility guidelines. This may have contributed to the team members' view that UX issues were the responsibility of the PO, UX designers and UI designers, and not themselves. Figure 11 is derived from Figure 10 and focuses on the agile development team alone. Where double-headed arrows appear in this diagram, detailed inspection shows that exchanges are restricted to clarification of *How* and some *What* information resulting in the translation of mock-ups and guidelines into task details.This emphasises the team members' passive consumption of UX information. A lack of engagement by developers with early UX discussions has been found to compromise technical qualities of the system (Plonka et al., 2014)

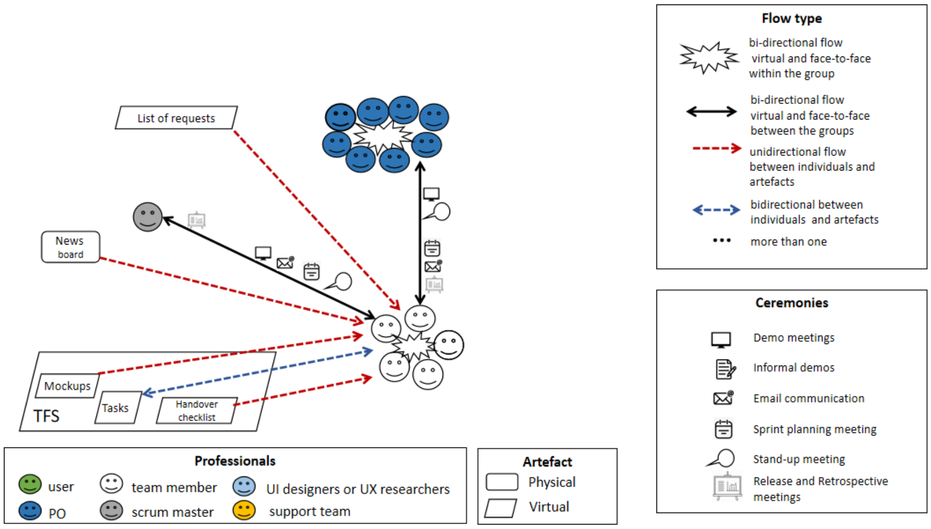

List of requests
News board
Mockups
Tasks
Handover checklist
TFS
Professionals
user
PO
team member
scrum master
UI designers or UX researchers
support team
Artefact
Physical
Virtual
Flow type
bi-directional flow virtual and face-to-face within the group
bi-directional flow virtual and face-to-face between the groups
unidirectional flow between individuals and artefacts
bidirectional between individuals and artefacts
more than one
Ceremonies
Demo meetings
Informal demos
Email communication
Sprint planning meeting
Stand-up meeting
Release and Retrospective meetings

Figure 11 UX Information flow focusing on the agile team, based on Fig 10. (Note: team members' input of UX information relates to translation of mock-ups and guidelines into code, focused on the How, rather than active engagement with What or Why)

Third, TFS contained most of the persistent information about UX. However, team members struggled to track UX information through it, and UX artefacts were often 'hidden' as attachments to tasks. If information is not visible or is difficult to collate, extra cognitive effort is needed to search for and gather it before use (Scaife and Rogers, 1996), and this can lead to mistakes. Kashfi et al. (2017) and Ferreira et al. (2012) also found that software practitioners struggle to keep UX information visible throughout software development. Virtual software management environments such as TFS are commonly used by agile teams (VersionOne, 2019), but they can be difficult to manage because of the wide choice of settings available (Akman et al., 2016).

Taken together, these observations indicate that UX information flow was complex and carried a high cognitive load, and that the structure of the commonly-used environment TFS contributed to that complexity. In addition, the more abstract levels of UX information relating to the user perspective were communicated verbally and often informally, which suggests a potential breakdown in understanding. Comparisons with the literature suggests that these findings are also relevant beyond the studied setting.

5.3 Recommendations to support agile teams in making better use of UX information

Based on our findings that are relevant beyond the studied setting, we suggest three areas for potential improvement in the use of UX information by agile teams:

1. make the *Why* and *What* of UX information from the users' point of view more explicit and visible to team members;
2. increase the level of engagement of team members with UX information; and,
3. decrease the complexity for team members of identifying, collating and hence tracking UX information.

Combining the discussion above and related literature we propose the following recommendations to address these areas:

5.3.1 Recommendation 1 – Create an informative workspace to make Why and What information more visible (addressing areas 1 and 2)

Although agile team members are aware of the importance of UX (Kollman et al, 2009), their mindset is mostly focused on coding (Ardito et al., 2014; Kuusinen, 2015) and an overall view of UX is uncommon in agile settings (Cajander et al., 2013; Bordin and Angeli, 2016), making it difficult for developers to engage with UX. This is exacerbated because agile and UX artefacts are often completely different and with little integration of information (Larusdottir et al., 2017; Garcia et al. 2019). A UX informative workspace that displays UX artefacts and end-user feedback could help the whole team remain engaged with UX, and promote discussion of UX at different levels (i.e. abstract and concrete perspectives). Making UX information visible is easier than trying to integrate information and artefacts.

Informative workspaces are not new in the agile context and have been used by UX designers working with agile teams; however, we are suggesting an informative workspace specifically to keep the user perspectives of *Why* and *What* aspects more visible, i.e. to make user feedback and user research part of the workspace for the team. Giving prominence to user research feedback impacts positively on the development of the team's expertise regarding users, has a motivational effect on developers (Sach, 2013), and supports team members in taking decisions for future product features or improvements (González-Torres et al., 2013; Paredes et al., 2014).

5.3.2 Recommendation 2- Encourage those who engage directly with users to take a more active role in facilitating the team's understanding of Why and What (addressing areas 1 and 2)

Agile practices often encourage end-user participation during the development process and several authors recognise that the involvement of end-users brings benefits and quality to product development (Cajander et al., 2013; Abelein and Paech, 2015), but this can be hard to achieve. Even when end-users are involved, agile teams may lack information about the product's motivation and its impact in the end-users' daily activity (Bordin and Angeli, 2016; Kashif et al., 2017), but when other professionals facilitate communication with end-users, their needs become advisory rather than integral to the team's work (Bordin and Angeli, 2016; Ardito et al., 2014).

To overcome the difficulties perceived in involving end users, and building on the common practice of having an intermediary, e.g. PO or UI designer, we suggest that those who engage directly with users take a more active role in helping team members to understand the users' motivations and goals.

5.3.3 Recommendation 3- Create guidelines to establish clear and simple structures to organise UX information in software tools (addressing area 3)

The effort required to acquire information using a virtual tool increases if navigation structures are complex, deep or disjointed (referred to as virtual distance in Bjarnason and Sharp, 2017), and traceability becomes difficult when information is dispersed (Kashfi et al., 2017). Liskin (2015) investigated impediments to artefact use in requirements communication and concluded that handling multiple artefacts is challenging and leads to manual translation effort and inconsistencies. Our DiCoT analysis indicates similar problems with UX information. Integrated software development tools are widely used by agile teams and are important to support the team sharing UX and software information (da Silva et al., 2018). To improve the situation with UX information we suggest the following:

1. encourage designers of these tools to support an integrated vision of UX. Most tools to support agile working still treat agile and UX as separate activities, which can lead to barriers that prevent UX from being seen as a cross-cutting concern (da Silva, 2018);

2. create guidelines or good practices that team members can use for linking and organising UX information within the tools, e.g. using information templates (Liskin, 2015);

3. create markers that highlight different types of information and differentiate UX information from other types, e.g. using different colours, highlights or keywords as tags. Each team could then devise a suitable UX vocabulary to give meaning to the tags;

4. keep UX artefacts in a central repository and reference them via links. This would help maintain consistency, avoid duplication of UX artefacts, and enhance UX traceability (Kashfi et al., 2017).

6. Study Validity

We discuss the trustworthiness of our work from the perspectives of *threats to validity in flexible designs*, *bias and rigour* and *generalizability* based on Robson and McCartan (2016). We identify relevant issues and discuss any mitigations below.

6.1 Threats to validity in flexible designs

The *description* of our study covers the setting details and the steps we followed. To guarantee the quality of our descriptions, we took into account data gathered from different sources, e.g. audio recording and hand-written notes. The methodology is reported in detail, supporting replication in other studies.

Our data *interpretation* was guided by the DiCOT framework and UX models which avoided some bias from the researchers' points of view. We chose two complementary UX models, Hassenzahl's model (2018) and Garrett's framework (2010), to mitigate any disconnected interpretation that could be arise from multiple researchers. Using these established approaches in our analysis provides a frame of reference for readers. The participation of three researchers avoided conclusions based on a single interpretation. Each step of the analysis was revised by the others through discussion of the results, findings and their implications. We did not adopt any *theory* to give a prior conception about the research.

6.2 Bias and rigour

The issues of *bias and rigour* may be found in research involving people observation, such as ethnographic studies. To mitigate such issues, we first sought to establish a *prolonged involvement* in the fieldwork by keeping close contact with the agile team members and other collaborators for three months. The development of a trusting relationship helped us to collect data from different perspectives and also observe the team work in different ceremonies, meetings and daily activities. We also had access to documents used by the team. The significant amount of gathered data supported *data triangulation* which gave us a better confidence in our interpretation. In addition, *debriefing sessions* were regulary conducted where the data and its interpretation were discussed between all in order to limit any influence of individual opinion. And we conducted regular *member checking* meetings with the SM to verify whether our interpretations were correct.

Reliability issues concern the methods and practices used to produce consistent results. In our study, we followed a set of procedures to guarantee data integrity (i.e. *audit trail*). We maintained a backup of all the audio recordings, photographs, notes in digital format and documents in an external hard drive. After each observation session or interview, the first author listened to the audio recordings and analysed the photographs, documents and the hand-written notes to create a memo document containing the initial data interpretation.

6.3 Generalizability in flexible designs

Internal generalizability refers to the generalizability of conclusions within the setting studied; the studied team represented one instance of several teams working on the same project who undertook similar activities using the same process. We believe that the results could be extend to other teams of the same organisation.

For *external generalizability*, there could be a limitation by focusing on one agile team. However, we compared the key results regarding artefacts and UX information with others' published findings. The consistency of our results with the ones presented in other studies gives us confidence that our findings are applicable more widely. The tools and artefacts adopted by the team are commonly used in other settings, and the team's product is a version of Moodle, a tool used by thousands of educational institutions worldwide. Detailed descriptions of team work are provided to allow others to explore whether the findings are applicable to another team.

7. Conclusions

In this paper, we have presented the results of an ethnographic study to understand the collaborative activities of an agile software development team with a specific focus on UX information. The distributed cognition framework for small teams (DiCoT) guided us in data collection and analysis. To examine the type of UX information available in the team's workplace, we drew on two complementary models of UX: Garrett's framework and Hassenzahl's content-oriented model. We have not found any other reported study that combines these analytical lenses. Although this study focused only on one team, the findings resonate with reported studies of other teams in different contexts, which provides confidence that aspects of our findings and the recommendations arising from them will be useful to others.

The study results in four main conclusions:

1. Agile team members are largely passive receivers of UX information and commonly have no active role in its creation. This reduces the teams' engagement with the information and may affect their shared understanding of UX; a lack of engagement in UX discussions may also compromise technical qualities of the system.
2. The commonly-used agile artefacts of user stories and mock-ups capture concrete UX information relating to *How* the user interacts with the product, and functional specifications about (rather than the user perspective on) *What* the product will do. As these artefacts are central to an agile team's work, this may lead to an overemphasis within a team's cognitive system on user interaction rather

than user needs or goals.

3. Information that focuses on the user perspective, relating to users' goals (*Why)* or *What* they want to do with the product, is not captured in artefacts and is communicated mostly in meetings and one-to-one verbal discussions. This results in multiple transformations of information which may lead to inconsistencies and misunderstandings of the users' perspective by the team, particularly if this information is not widely discussed.
4. Tracking UX information through the cognitive system supporting the studied team was complex and time-consuming, requiring cognitive effort to complete. This was mainly due to the structure of UX information within the key information buffer TFS, which is widely adopted by agile teams. This complexity may affect productivity and result in agile team members having only a partial understanding of the product's UX.

Taken together, these findings indicate a potential communication breakdown in the cognitive system that supports UX activities, and we suggest three areas which would benefit from improvement: making the user perspective, specifically the *Why* and *What* of UX information, more explicit and visible to team members; increasing the level of engagement of team members with UX information; and, decreasing the complexity for team members to identify, collate and hence track UX information. To address these areas we recommend the use of a UX informative workspace to promote better understanding of the users' point of view, and engagement with and visibility of *Why* and *What* UX information; more active attempts to communicate the users' perspective of *What* and *Why* to team members; and development of guidelines for clear organisation of UX information in software development environments such as TFS.

Potential future work in this area includes developing a more sophisticated model of UX information, e.g. by synthesising Hassenzahl's model and Garrett's framework; implementing the recommendations and evaluating the results; and investigating further the analysis of virtual artefacts and online development environments using distributed cognition principles. Work is increasingly conducted within a virtual environment rather than a physical one, and analytical frameworks need to adapt to take this into account.

8. Acknowledgments

The authors thank our collaborators for their participation in the study. We also acknowledge the financial support from grant#2017/03397-0, São Paulo Research Foundation (FAPESP); grant 313312/2019-2 (CNPq); and Agile Business Consortium, Agile Research Network.

References

Abdullah, N.N.B., Sharp, H., Honiden, S., 2010. Communication in context: A stimulus-response account of agile team interactions, in: Sillitti, A., Martin, A., Wang, X., Whitworth, E. (Eds.), Agile Processes in Software Engineering and Extreme Programming, Springer Berlin Heidelberg, Berlin, Heidelberg. pp. 166–171.

Abelein, U.; Paech, B., 2015. Understanding the Influence of User Participation and Involvement on System Success – a Systematic Mapping Study. Empirical Software Engineering, vol. 20, pp. 28–81.

Akman, S., Ozmut, M., Aydın, B., Gokturk, S., 2016. Experience report: implementing requirement traceability throughout the software development life cycle. Journal of Software: Evolution and Process, vol 28, n. 11, pp. 950–954.

Ardito, C., Buono, P., Caivano, D., Costabile, M.F., Lanzilotti, R., 2014. Investigating and promoting UX practice in industry: An experimental study. International Journal of Human-Computer Studies, vol 72, n. 6, pp. 542–551.

Bjarnason, E., Sharp, H., 2017. The role of distances in requirements communication: a case study. Requirements Engineering, vol 22, pp. 1–26.

Beyer, H., Holtzblatt, K., 1998. Contextual Design. Morgan Kaufmann, San Francisco, CA.

Bordin, S., Angeli, A.D., 2016. Focal points for a more user-centred agile development. In: Sharp, H., Hall, T. (Eds.), Agile Processes, in Software Engineering, and Extreme Programming, Springer International Publishing, pp. 3–15.

Brhel, M., Meth, H., Maedche, A., Werder, K., 2015. Exploring principles of user-centered agile software development: A literature review. Information and Software Technology, vol 61, n. C, pp. 163 – 181.

Brown, J.M., Lindgaard, G., Biddle, R., 2011. Collaborative events and shared artefacts: Agile interaction designers and developers working toward common aims, In: Proceedings of the 2011 Agile Conference, pp. 87–96.

Buchan, J., 2014. An Empirical Cognitive Model of the Development of Shared Understanding of Requirements. Communications in Computer and Information Science, vol 432 CCIS, pp. 165–179.

Cajander, Å., Larusdottir, M., Gulliksen, J., 2013. Existing but Not Explicit - The User Perspective in Scrum Projects in Practice. In: Kotzé P., Marsden G., Lindgaard G., Wesson J., Winckler M. (eds) Human-Computer Interaction – INTERACT 2013. INTERACT 2013. Lecture Notes in Computer Science, vol 8119. Springer, Berlin, Heidelberg.

Chamberlain, S., Sharp, H., Maiden, N., 2006. Towards a Framework for Integrating Agile Development and User-Centred Design. In: Abrahamsson P., Marchesi M., Succi G. (eds) Extreme Programming and Agile Processes in Software Engineering. XP 2006. Lecture Notes in Computer Science, vol 4044. Springer, Berlin, Heidelberg.

Choma, J., Zaina, L.A.M., Beraldo, D., 2016. UserX story: Incorporating UX aspects into user stories elaboration, in: Kurosu, M. (Ed.), Human-Computer Interaction. Theory, Design, Development and Practice, Springer International Publishing. pp. 131–140.

Cohn, M., 2009. User stories applied: for agile software development. Pearson Education, Inc. Boston, MA.

Deshpande, A., Sharp, H., Barroca, L., Gregory, P., 2016. Remote Working and Collaboration in Agile Teams. In:

Proceedings of the International Conference on Information Systems - Digital Innovation at the Crossroads, pp. 11–14 Dublin, Ireland.

Ferreira, J., Sharp, H., Robinson, H., 2012. Agile development and user experi- ence design integration as an ongoing achievement in practice, in: 2012 Agile Conference, pp. 11–20.

Fox, D., Sillito, J., Maurer, F., 2008. Agile methods and user-centered design: How these two methodologies are being successfully integrated in industry, in: Agile 2008 Conference, pp. 63–72.

Furniss, D., Blandford, A., 2006. Understanding emergency medical dispatch in terms of distributed cognition: A case study. Ergonomics, vol 49, pp. 1174–1203.

Garcia, A., da Silva T. S., Selbach Silveira, M., 2017. Artifacts for agile user-centered design: A systematic mapping. In: Proceedings of the 50th Hawaii International Conference on System Sciences.

Garcia A., da Silva T.S., Silveira M.S., 2019. Artifact-Facilitated Communication in Agile User-Centered Design. In: Kruchten P., Fraser S., Coallier F. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2019. Lecture Notes in Business Information Processing, vol 355. Springer, Cham.

Garrett, J.J., 2010. The Elements of User Experience: User-Centered Design for the Web and Beyond. 2nd ed., New Riders Publishing, Thousand Oaks, CA, USA.

Gonzalez-Torres, A., Garcıa-Penãlvo, F.J., Theron, R., 2013. Human–computer interaction in evolutionary visual software analytics. Computers in Human Behavior, vol 29, pp. 486 – 495.

Gray, C.M., 2016. “It’s more of a mindset than a method”: UX practitioners’ conception of design methods. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, ACM, New York, NY, USA. pp. 4044–4055.

Hassenzahl, M., 2010. Experience design: technology for all the right reasons. Morgan Claypool, San Francisco.

Hassenzahl M.,2018. The Thing and I (Summer of ’17 Remix). In: Blythe M., Monk A. (eds) Funology 2. Human–Computer Interaction Series. Springer, Cham.

Hoadley, C.M., Kilner, P.G., 2005. Using Technology to Transform Communities of Practice into Knowledge-Building Communities, SIGGROUP Bulletin, vol 25, n. 1, pp. 31-40.

Hollan, J., Hutchins, E., Kirsh, D., 2000. Distributed cognition: Toward a new foundation for human-computer interaction research. ACM Transactions on Computer-Human Interaction, vol 7, n. 2, pp. 174–196.

Hutchins, E., 1995. Cognition in the wild. A Bradford Book.

Jurca, G., Hellmann, T.D., Maurer, F., 2014. Integrating agile and user-centered design: A systematic mapping and review of evaluation and validation studies of Agile-UX, In: Proceedings of the 2014 Agile Conference, pp. 24–32.

Kashfi, P., Nilsson, A., Feldt, R., 2017. Integrating user experience practices into software development processes: implications of the UX characteristics. PeerJ Computer Science 3:e130.

Kollman, J., Sharp, H. and Blandford, A. (2009) ‘The role of user experience practitioners on agile project’ in Proceedings of Agile 2009

Kuusinen, K., 2015. Task allocation between UX specialists and developers in agile software development projects, in: Abascal, J., Barbosa, S., Fetter, M., Gross, T., Palanque, P., Winckler, M. (Eds.), Human-Computer Interaction–INTERACT 2015, Springer International Publishing.

Lanubile, F., Calefato, F., Ebert, C., 2013. Group awareness in global software engineering. IEEE Software, vol 30, pp. 18–23.

Larusdottir, M., Gulliksen, J., Cajander, A., 2017. A license to kill – Improving UCSD in Agile development. Journal of Systems and Software, vol 123, pp. 214–222.

Liskin, O., 2015. How Artifacts Support and Impede Requirements Communication, In: Proceeding of the 21st International Working Conference, REFSQ 2015, Essen, Germany, March 23–26, pp. 132–147.

Liskin, O., Schneider, K., Fagerholm, F., Munch, J., 2014. Understanding the role of requirements artifacts in Kanban. In: Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering, ACM, New York, NY, USA. pp. 56–63.

Mangalaraj, G., Nerur, S., Mahapatra, R., Price, K.H., 2014. Distributed cognition in software design: An experimental investigation of the role of design patterns and collaboration. MIS Q, vol 38, pp. 249–274.

Matta, M., Marchesi, M., 2015. Understanding approval rating of agile project management tools using twitter, in: 2015 10th International Joint Conference on Software Technologies (ICSOFT), pp. 1–6.

McInerney, P., 2017. UX in agile projects: Taking stock after 12 years. Interactions, vol 24, pp. 58–61.

Paredes, J., Anslow, C., Maurer, F., 2014. Information visualization for agile software development. In: Proceedings of the 2014 Second IEEE Working Conference on Software Visualization, pp. 157–166.

Plonka, L.; Sharp, H.; Gregory, P., Taylor, K., 2014. UX design in agile: a DSDM case study. In: Cantone G., Marchesi M. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2014. Lecture Notes in Business Information Processing, vol 179. Springer, Cham.

Robson, C.; McCartan, K., 2016. Real world research: a resource for users of social research methods in applied settings, 4th edition, Wiley.

Rubin, K.S., 2012. Essential Scrum: A Practical Guide to the Most Popular Agile Process. 1st ed., Addison-Wesley Professional.

Sach, R., 2013. The Impact of Feedback on the Motivation of Software Engineers. PhD thesis. The Open University.

Salah, D., Paige, R.F., Cairns, P., 2014. A systematic literature review for agile development processes and user centred design integration. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, pp. 5:1–5:10, ACM, New York, NY, USA.

Scaife, M., Rogers, Y., 1996. External cognition: How do graphical representations work?. International Journal of Human

Computer Studies, vol 45, n. 2, pp. 185–213.

Schon, E. M., Thomaschewski, J., Escalona, M. J., 2017. Agile requirements engineering: A systematic literature review. Computer Standards Interfaces, vol 49, pp. 79 – 91.

Sellberg, C., Lindblom, J., 2014. Comparing methods for workplace studies: a theoretical and empirical analysis. Cognition, Technology and Work, vol 16, n. 4, pp. 467–486.

Sharp, H., Dittrich, Y., de Souza, C.R.B., 2016. The role of ethnographic studies in empirical software engineering. IEEE Transactions on Software Engineering, vol 42, n. 8, pp. 786–804.

Sharp, H., Giuffrida, R., Melnik, G., 2012. Information flow within a dispersed team: a distributed cognition perspective. In: Wohlin C. (eds) Agile Processes in Software Engineering and Extreme Programming. XP 2012. Lecture Notes in Business Information Processing, vol 111. Springer, Berlin, Heidelberg.

Sharp, H., Robinson, H., 2008. Collaboration and Co-ordination in mature eXtreme Programming teams. International Journal of Human-Computer Studies, vol 66, n. 7, pp. 506–518.

Sharp H., Robinson H., 2006. A Distributed Cognition Account of Mature XP Teams. In: Abrahamsson P., Marchesi M., Succi G. (eds) Extreme Programming and Agile Processes in Software Engineering. XP 2006. Lecture Notes in Computer Science, vol 4044, pp. 1–10. Springer, Berlin, Heidelberg.

da Silva, T.S., Martin, A., Maurer, F., Silveira, M., 2011. User-centered design and agile methods: A systematic review, In: Proceedings of the 2011 Agile Conference, pp. 77– 86.

da Silva, T.S., Silveira, M.S., Maurer, F., Silveira, F.F., 2018. The evolution of agile UXD. Information and Software Technology, vol 102, pp. 1 – 5.

Version One, 2019. 14th State of Agile Report, Available at: https://stateofagile.com/#ufh-c-7027494-state-of-agile.

Wale-Kolade, A., Nielsen, P.A., Päivärinta, T., 2013. Usability Work in Agile Systems Development Practice: A Systematic Review. In: Linger H., Fisher J., Barnden A., Barry C., Lang M., Schneider C. (eds) Building Sustainable Information Systems, pp. 569–582. Springer, Boston, MA

Wright, P.C., Fields, R.E., Harrison, M.D., 2000. Analyzing Human-Computer Interaction as Distributed Cognition: The Resources Model. Human–Computer Interaction, vol 15, n. 1, pp. 1–41.