2013

Agile User Experience Design: A Design Science Enquiry

Sisira Adikari
University of Canberra, Sisira.Adikari@canberra.edu.au

Craig McDonald
University of Canberra, craig.mcdonald@canberra.edu.au

John Campbell
University of Canberra, john.campbell@anu.edu.au

Information Systems: Transforming the Future

ACIS 2013
RMIT MELBOURNE

24th Australasian Conference on Information Systems, 4-6 December 2013, Melbourne

Proudly sponsored by

ACIS 2013 Principal Sponsor

Agile User Experience Design: A Design Science Enquiry

Sisira Adikari, Craig McDonald and John Campbell
School of Information Systems and Accounting
Faculty of Business, Government and Law
University of Canberra
ACT 2601, Australia
Email: {sisira.adikari, craig.mcdonald, john.campbell}@canberra.edu.au

Abstract

This paper presents a qualitative analysis of an information systems design study within the context of agile software development based on conceptually different two design approaches: Current Agile Process (CAP) and Enhanced Agile Process (EAP). Eight agile software professionals and one user experience designer were recruited from the industry to form two small agile teams, where one agile software professional shared the product owner role in both teams. Each team undertook agile software product development based on exclusively one of the two conceptually different design approaches, CAP and EAP. Both teams used the same suite of user stories presented by the product owner. The progress of product development was assessed using four evaluation approaches: observation data comparison, debrief data comparison, individual system evaluation and comparative system evaluation. This paper presents the results of observation data comparison only. The results suggest that agile software development team that followed the enhanced agile process had more user experience focus in the design and the outcome of the product delivery proved to be somewhat richer in user experience.

Keywords

Agile Software Development, Design Science, User Experience Design

INTRODUCTION -

Agile Software Development (ASD) has been recognised as an effective approach for gaining wider acceptance of software products by the industry (West and Grant, 2010). Specifically, ASD emphasises the importance of communication and collaboration among stakeholders in identifying, exploring and addressing issues in a team culture of knowledge sharing, care and ensuring mutual trust (Adkins, 2010). Moreover, agile principles are based on increased collaboration and social interaction, emphasising people over processes, working software over comprehensive documentation, and responding to change over following a fixed plan. The latter two principles intended to discourage lock-in requirements with up-front plans and documentation. In a classical software engineering book, Boehm (1981) has clearly pointed out that more the design is determined up-front; if needed; it will be more expensive to change later in the software development. Moreover, when such change is necessary, all effort of design up-front is wasted. In contrast to ASD, User Experience Design (UXD) advocates an up-front user research activity, design, and evaluations of design artefacts with end-users. These intensive design methods are commonly referred to as Big Design Up-Front (Slimick, 2007) and the emphasis is to create a 'big' design before coding and quality assurance testing takes place. Accordingly, UXD is not a natural fit with ASD; hence there is a reluctance to integrate UXD with Agile. Comparative differences between UXD and ASD are:

- UXD activities advocate that certain design products are required in support of communication with developers, while ASD seek minimal documentation.
- UXD encourages the team to understand their users as much as possible before the software development, whereas ASD is largely against an up-front period of investigation at the expense of writing code.

As reported in the literature, there have been many proposals, suggestions as well as empirical research to explain how UXD can be integrated into ASD. Many of these references either focus on the common iterative process behaviour of UXD and ASD as the common ground of integration or point out the importance of having an 'iteration 0' for UXD ground preparation and planning activities. Importantly, a common viewpoint shared by many of the authors is that UCD needs sufficient time and resources to conduct user research before development iterations begin. Accordingly, UCD integration into ASD is not that simple because of the additional time and resource requirements and is likely to impose a potential challenge. Moreover, agile software processes seek to follow an evolutionary approach to define requirements during the course of analysis and design, which is known as Just-In-Time (JIT) requirements analysis.

In this paper, we report the results of an information systems design study where an integrated Agile-UXD approach, which was based on JIT analysis and design to engage UXD professionals within the agile team for carrying out JIT analysis and design tasks. We used design science research as the primary method of enquiry. This integrated approach intended to provide only required details of UXD information as needed to support the analysis and design during agile iterations. The objective was to provide only sufficient UXD information to support the popular agile JIT analysis and design so that the UXD perspective could be considered without overloading existing agile practices.

DESIGN SCIENCE RESEARCH – A BRIEF OVERVIEW

The Information Systems (IS) literature clearly shows the recognition of Design Science Research (DSR) as an effective and alternative research paradigm for Information Systems Design (Hevner et al., 2004), (March and Smith, 1995). The acceptance of DSR for IS research is further emphasised in recent publications such as (Iivari, 2007), (Kuechler and Vaishnavi, 2008). There are many definitions of DSR but the general consensus of these definitions is about creating and evaluating artefacts as solutions to relevant design problems. For example, one of the recent definitions of DSR states that it is a research that invents a new purposeful artefact to address a generalised type of problem and evaluates its utility for solving problems of that type (Venable and Baskerville 2012). Hevner et al. (2004) outlined the initial framework of the DSR for IS context. Based on the initial framework, Hevner (2007) presented a comprehensive view of DSR by means of three research cycles (see Figure 1) to show a detailed process view for performing DSR.

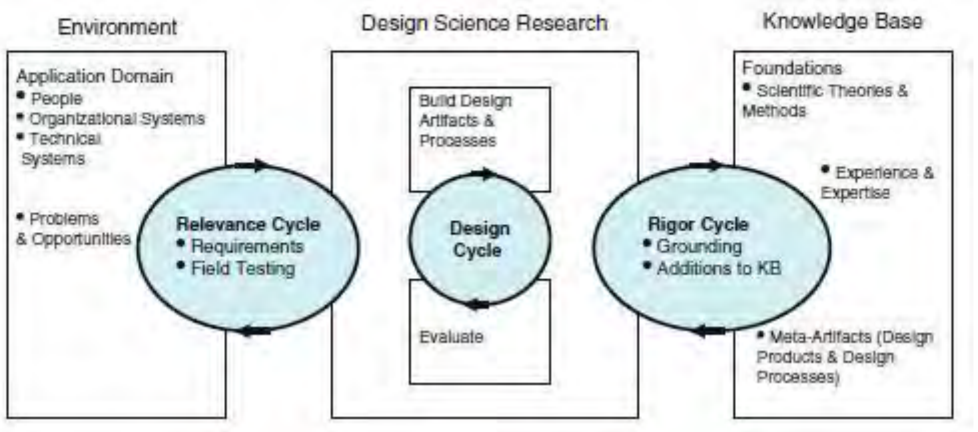

Figure 1: Design science research cycles (Hevner and Chatterjee 2010, p.16)

The aim of the DSR is to improve the environment with new and innovative artefacts (Simon, 1996) for contextual problems. In Figure 1, there are three main components of DSR namely; environment, design science research, and knowledge base. The ***Environment*** represents the context where the application domain (people, organisational systems, technical systems) exists. The ***Knowledge Base*** consists of scientific foundations, experience and expertise, and meta-artefacts (design products and design processes). The ***Design Science Research*** is the main research process which carries out two core activities: building design artefacts and processes, and evaluating design artefacts and processes. The DSR process is carried out in three inherent research cycles namely; relevance cycle, rigor cycle, and design cycle.

The relevance cycle connects the contextual environment with the design science research to provide relevant contextual information for DSR activities, and it also delivers and realises the DSR outcome in the context. The rigor cycle connects the knowledge base with DSR to provide the existing knowledge for DSR activities and also feedbacks the knowledge base with updated knowledge after DSR activities. The internal design cycle iterates between the two core DSR activities to build and evaluate design artefacts and processes.

The outcome of a DSR project is broadly defined as five types of IT artefacts namely; constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), instantiations (implemented and prototype systems) and better design theories (Hevner and Chatterjee, 2010, p.6). Importantly, the last definition emphasises the clear focus of DSR in generating better design theories.

USER EXPERIENCE AND AGILE USER EXPERIENCE DESIGN

Although the academic literature and the industry terminology widely use the term 'user experience design', it has been clearly pointed out that one cannot design a user experience, only design for a user experience (Rogers, et al., 2011, p.14). Accordingly, the agile User Experience (UX) design refers to 'designing for user experience' in Agile Software Development (ASD) context. Another important distinction is the difference between UX and User-Centred Design (UCD). UX is the extended usability which evolved from classical usability and the main focus of UX is distinctive in creating positive UX mainly by means of pleasure, joy, excitement, fun, attitudes, emotions and added values when the user interacts with an artefact. The multidisciplinary nature of UX has led to different perspectives and definitions of UX (Roto et al., 2011), (Hiroyuki, 2013). The latest ISO standard ISO 9241-210 (2010) clearly makes a distinction between usability and UX (ISO, 2010). According to the standard, Usability is the "extent to which a product, system or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. The standard states UX as "A person's perceptions and responses resulting from the use and/or anticipated use of a product, system or service" Analysing the standard ISO 9241-210 (2010) further, Bevan (2013) provides UX interpretation of usability as "extent to which a product, system or service can be used by a person to achieve a specific goal with effectiveness, efficiency and satisfaction in a specific context of use". This interpretation shows that usability as the satisfaction with the artefact (product, system or service) whilst UX as the satisfaction with the interactive experience as a result of gratification (the feeling of pleasure that comes when a need or desire is fulfilled) and assessment (extent to which gratification of a need or desire meets expectations).

The UX design in ASD has been widely discussed in the literature, and there are continuing discussions about how this integration can be achieved (Ferreira et al., 2012). These studies highlight the fact that UX aspects are not well addressed in ASD (Chamberlain et al., 2006), (Hussain et al., 2009). The UX design and ASD belong to two distinct disciplines with entirely different focus of software development. UX is central to interaction design where the main focus is creating positive experiences when the user interacts with an artefact (a product, system or service). On the other hand, ASD mainly focuses on developing working software according to customer requirements in short iterative and incremental development cycles. One speciality of the ASD is that product requirements and working software evolve during the software development iterations. Moreover, UX design consists of time-consuming and effort-intensive up-front research and activities such as contextual enquiries, interviews, definition of UX and usability goals, creation of personas and usage scenarios etc., resulting in extensive documentation. Such activities perceived to be Big Design Up-Front is the design philosophy in UX design. In contrast, ASD is light-weight, customer-oriented and a highly collaborative approach that follows a continual exploration of the business need as the basis to gather and refine software requirements to develop quality software (Adikari et al., 2013). The ASD follows Just-In-Time analysis and design to define product requirements during development iterations with minimal up front design elaboration leaving space for design changes later in software development. Agile software processes also discourage extensive up-front design (McInerney and Maurer, 2005). Accordingly, one of the important issues in integrating UX Design and ASD is associated with the different approaches they follow in terms of requirements analysis and design (Fox et al., 2008). Moreover, Adikari et al. (2013) reported that a key challenge faced by UX design for ASD is the building an in-depth understanding of the user and the context of use in a manner that adds business value for ASD process and activities.

INTEGRATION OF USER EXPERIENCE DESIGN AND AGILE SOFTWARE DEVELOPMENT – CHALLENGES AND APPROACHES

In the literature, many authors have reported how the UX design and ASD can be integrated together to deliver more usable, useful and desirable products. One of the important issues identified is the lack of user perspective and the user involvement in ASD processes. The customer orientation nature of the ASD emphasises customer as the main source of information for product requirements, and there is no direct user involvement as such. Moreover the agile manifesto does not clearly demand the engagement of end-users as customers (Wolkerstorfer et al., 2008). Many authors have identified the distinction between the user and the customer in ASD context highlighting that customers only have a limited understanding of the users needs' simply because they are not the actual users of the software to be developed (Bayer et al., 2004), (Kautz, 2009). Chamberlain et al. (2006) inform the fact that ASD methods are largely against an up-front period of investigation. They also propose five principles for integrating UCD and Agile development effectively. These principles have been imposed on topics namely: user involvement, collaboration and culture, prototyping, project lifecycle and project management. Lee and McCrickard (2007) argue that a joint approach of traditional usability and traditional software development is difficult because agile methods do not support any kind of comprehensive overview of the usability aspects which can become a bottleneck in the overall development process. Analysing Scrum Agile context, Budwig et al. (2009) see coordination and collaboration as an avenue of solution towards combining the UX and ASD. They proposed scheduling UX teams to work one or two sprints ahead of the development teams and importantly, forming a separate UX agile team with its own product owner and product backlog. However, the importance of 'one team' concept in agile context has been highlighted by (2010) to emphasize that the whole ASD team should take the responsibility for whole team's work and UX professionals should be recognized as full team members of the ASD. Such an approach will provide the necessary UX knowledge for product development process, otherwise, UX expertise is simply ignored, under the assumption that the team's product owner can provide all the user interface knowledge required. Although the argument by Beyer is quite valid, according to the literature, there is little guidance on the engagement of UX professionals in ASD (Silva et al., 2011).

There are a number of design approaches proposed by many authors in support of combining UX design and agile software development. For example, Sy and Miller (2008) proposed an approach where UX designers work in an iteration 'zero' to begin with and then continue the UX design work one iteration ahead of the agile developers. In an early research study on up-front inteaction design in agile context, Ferreira et al. (2007) reported that although there was a conviction of advantages in up-front interaction design, integrating interaction design in agile context was primarily focused on producing working software rather than early design concepts. Moreover, it was not as clear as to whether up- front interaction design was feasible in agile context that adds value to the agile software development practice. Authors Haikara (2007) and Broschinsky and Baker (2008) advised the use of 'persona' as a design approach in agile context to extend the usability aspects towards product development. Adikari et al. (2011) proposed a research framework that assessed the impact of UCD on end products of ASD. In their pilot study, one user-centred designer was included in the ASD iterations to provide additional user-centric perspective for design analysis. Adikari et al. (2009) also proposed a design science based approach to integrating usability into agile requirements engineering. Their approach based on the concept 'Little Design Up-Front' (LDUF) – an approach providing *only required* details of UCD information *as needed* to support the analysis and design in agile iterations.

RESEARCH DESIGN

This research study consisted of two small-scale agile projects. The first project was based on the traditional Scrum agile methodology and the design approach followed was designated as Current Agile Process (CAP). The first project also served as the baseline reference to compare the second project which followed different design approach designated as Enhanced Agile Process (EAP). The EAP was an approach where UX aspects were purposefully integrated in the traditional agile process which was based on the Scrum agile methodology.

The first project was a typical agile project with three software development iterations. The research design of the first project is shown in Figure 2.

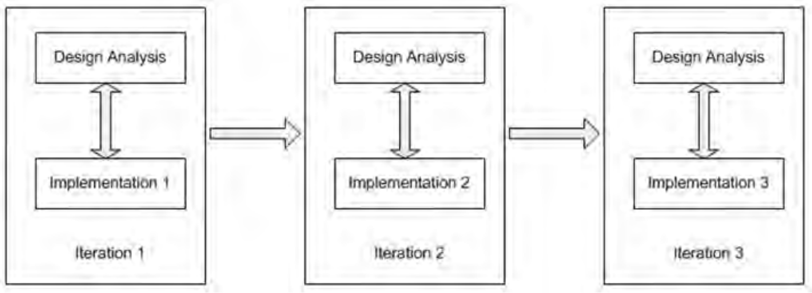

Figure 2: Research design – Agile project (CAP)

Research Process of the CAP Project

The agile team in the first study CAP consisted of a product owner, an agile coach and three agile software developers who worked together as a team for Just-In Time (JIT) design analysis. The product owner was an agile software professional who shared the product owner role in both CAP as well as EAP team. Product owner provided same suite of user stories of a real-world music management system for both teams. The CAP agile team worked very closely with the product owner to create and assess design artefacts in support of analysis, verification and validation.

The CAP study ran for three development iterations. The first iteration was specifically focused on requirements analysis and setting up the product backlog. The agile team worked under the guidance of the agile coach to produce working software. At the end of the first iteration, the agile team formally presented the first version of the working software to the product owner for assessment. In consultation and agreement with the product owner, the product backlog was updated and the second iteration was planned. The second and third iterations conducted in the same way as the first iteration based on similar agile settings and principles. At the end of the third iteration, the product owner formally assessed the final end product delivered by the agile team for sign-off.

The second ASD study followed the Enhanced Agile Process (EAP) that aimed at incorporating UX into traditional ASD. The research design of the first project is shown in Figure 3.

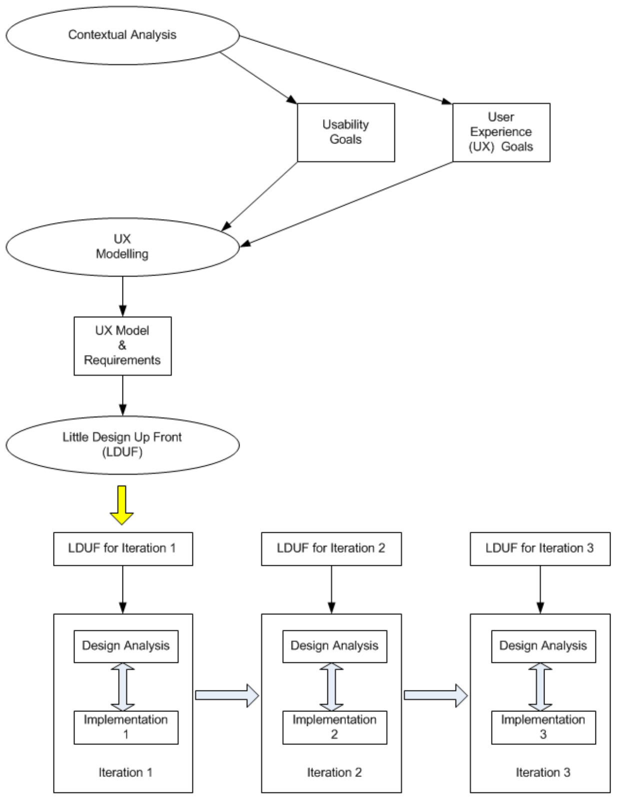

Figure 3: Research design – Agile project (EAP)

Research Process of the EAP Project

In the second study (EAP), one important exception was the engagement of a UX Designer in the agile team. Similar to the first study CAP, a product owner, an agile coach, two agile software developers and the UX designer worked together as a team for JIT design analysis. Importantly, there was one product owner for both studies to ensure a consistency in communicating product requirements to the team. The UX designer created Little Design Up-Front (LDUF) artefacts for design analysis. The agile team and the UX designer worked very closely with the product owner to create and assess design artefacts in support of analysis, verification and validation.

Little Design Up-Front (LDUF) is a design science research oriented process which was used to create design artefacts that contains only the sufficient and required information for design activities in ASD iterations. As shown in Figure 3, usability goals and UX goals were derived from the contextual analysis information. These usability goals and UX goals were then transformed into a UX model and requirements through the process 'UX modelling'. Subsequently, the UX model and requirements were used as the basis to create LDUF artefacts for all development iterations.

In the second study EAP, the contextual analysis, determining usability goals and UX goals, UX modelling, specifying the UX model and requirements, and creating LDUF artefacts were all conducted by the UX designer engaged in the Enhanced Agile Process (EAP) as part of the just-in-time design analysis.

The EAP study also ran for three development iterations. The first iteration was specifically focused on requirements analysis and setting up the product backlog. The agile team worked under the guidance of the agile coach to produce working software. At the end of the first iteration, the agile team formally presented the first version of the working software to the product owner for assessment. In consultation and agreement with the product owner, the product backlog was updated and the second iteration was planned. The second and third iterations conducted in the same way as the first iteration based on similar agile settings and principles. At the end of the third iteration, the product owner formally assessed the final end product delivered by the agile team for signed it off.

The EAP study also ran in a similar fashion to the first CAP study with one use experience designer engaged in the agile team to create LDUF artefacts as part of the design analysis for each of the development iteration. Throughout the software development, the UX designer also worked very closely with the agile team and the product owner to create and assess design artefacts in support of analysis, verification and validation. At the end of the third iteration, the product formally assessed the end product delivered by the agile team and signed it off.

DATA COLLECTION AND ANALYSIS

This section covers the data collection and data analysis of the research study based on the research design of data collection and analysis shown in Figure 4. This section also presents the findings of first data collection process only and discusses key points, implications and limitations.

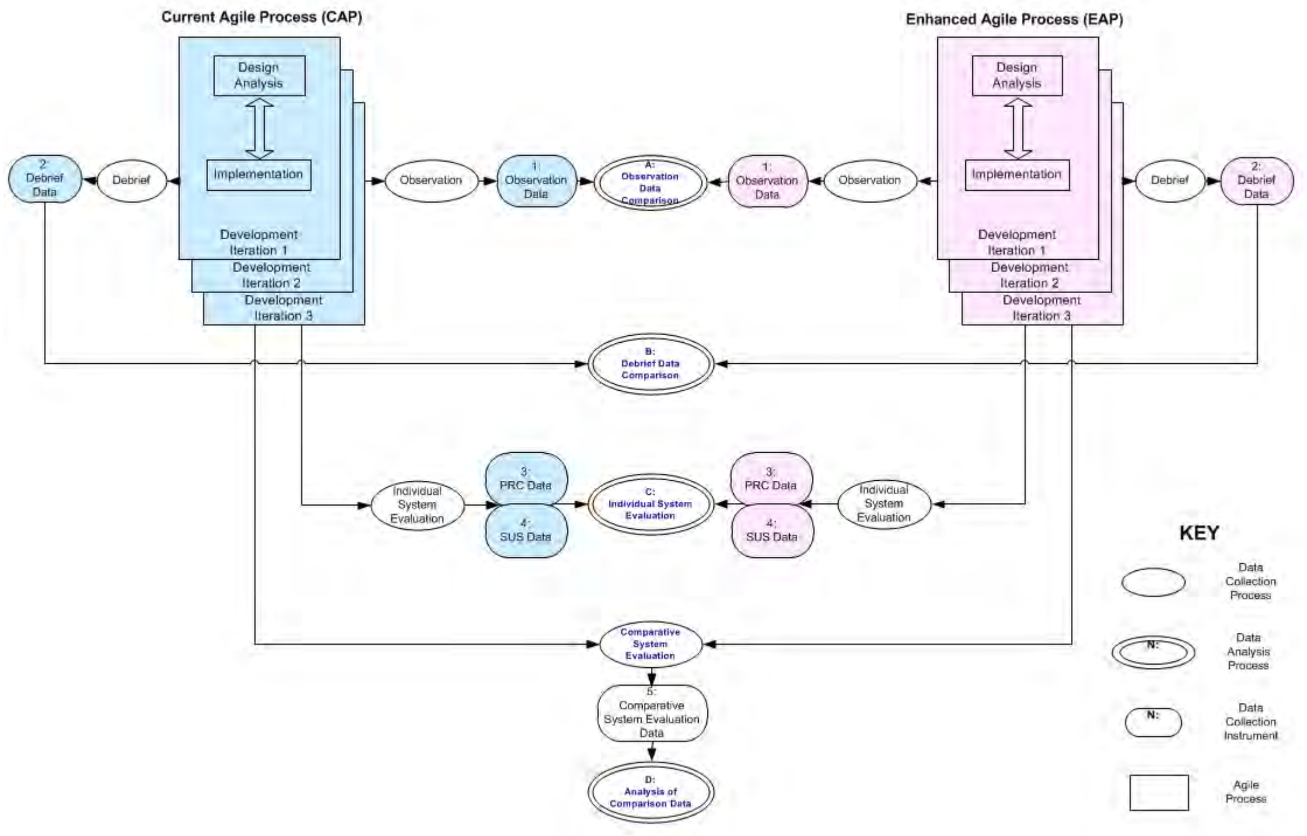
Current Agile Process (CAP)
Design Analysis
Implementation
Development Iteration 1
Development Iteration 2
Development Iteration 3
2: Debrief Data
Debrief
Observation
1: Observation Data
A: Observation Data Comparison
1: Observation Data
Observation
Enhanced Agile Process (EAP)
Debrief
2: Debrief Data
B: Debrief Data Comparison
Individual System Evaluation
3: PRC Data
4: SUS Data
C: Individual System Evaluation
3: PRC Data
4: SUS Data
Individual System Evaluation
Comparative System Evaluation
5: Comparative System Evaluation Data
D: Analysis of Comparison Data
KEY
Data Collection Process
N: Data Analysis Process
N: Data Collection Instrument
Agile Process

Figure 4: Research design –Data collection and analysis

As shown in Figure 4, the data analysis comprised of four data analysis processes namely: **A**: Observation data comparison, **B**: Debrief data comparison, **C**: Individual system evaluation, and **D**: Comparative system evaluation.

This section only shows the process and analysis of observation data collection. The aim of the observation data comparison was two-fold. First, it focused on gaining a deeper understanding of the software development processes in agile context. Next, it further explored how the individual team players work together as team, overcome software development problems, issues and challenges towards productive solutions. The observation data was collected by means of observing and recording the dynamics of the software development team in context, hence the nature of the data was qualitative. Next section explains the observation data comparison in detail.

OBSERVATION DATA COMPARISION

This section shows the findings and analysis of observation data comparison. The observation data is the raw data collected by observing the social interaction of individual team members in each of the ASD iteration.

The qualitative data collected during the observation data collection process was subjected to the coding process during which raw data were examined in order to identify meanings and ideas associated with segments of data. We applied open coding process to generate perspectives or themes of information from the raw data. During the open coding process, raw data were closely examined and conceptual codes were assigned to segments of data that show some thematic significance. The conceptual codes were developed based on existing literature in the fields of Human-Computer Interaction, Software Engineering and Project Management. These thematic codes were then compared to identify similarities, and differences. The similar codes were grouped together to form conceptual categories.

In this study, the observation data mainly constituted of conversations, agreements, concerns as well as varying subjective expressions. The following section describes in detail about the data analysis and comparison process conducted for observation data in CAP and EAP.

Vertical Observation Data Comparison

Figure 5 shows the design of the vertical observation comparison process which was conducted for data collection, analysis, and comparison.

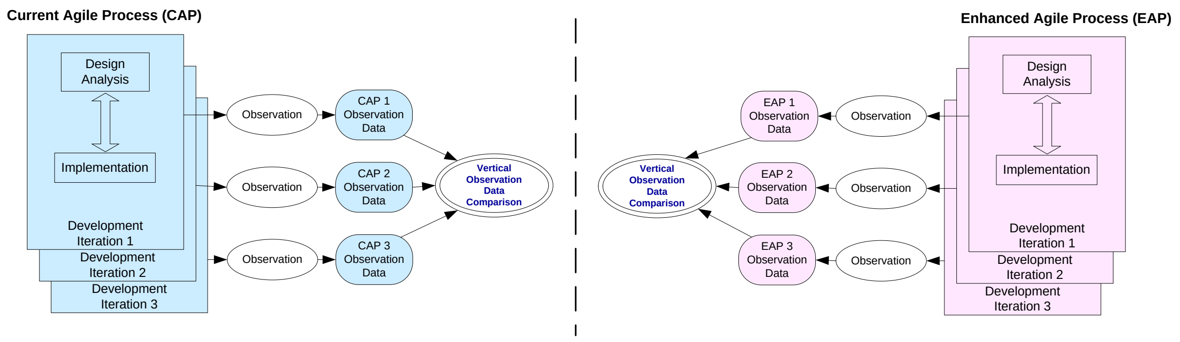

Figure 5: Vertical observation data analysis and comparison

The vertical observation data analysis and comparison process was conducted in two steps. In the first step, CAP 1, CAP 2, and CAP 3 observation data were compared collectively to identify similarities and differences of themes in observation data from each of the iteration. Following a similar approach, in step 2, EAP 1, EAP 2, and EAP 3 observation data were compared collectively to identify similarities and differences of themes in the observation data from of the each iteration.

Analysis CAP

As a result of the open coding process conducted on the observation qualitative data collected for each of the CAP iteration identified a number of themes. Table 6-1 shows the identified themes for all CAP iterations.

Table 1. Identified themes in CAP iteration 1, 2, and 3.

CAP Iteration - 1	CAP Iteration - 2	CAP Iteration - 3
Development Environment	Development Progress	Development Progress
Clarity of requirements	Product Requirements	Product Requirements
Development	Development	Development Planning
Whole system view	System View	Integration
Integration	Integration	
User aspects		

The themes that shared some commonality on a specific area of focus were categorised as ‘similar’, under the identified area of focus and a code of the conceptual category was given. The themes that did not share a conceptual commonality with other themes were considered as themes that occurred with no relevance hence ranked as ‘different’ without a conceptual category code. Table 2 shows the grouped themes and the formation of respective CAP conceptual categories.

Table 2. Conceptual categories - CAP iteration 1, 2, and 3.

CAP Iteration - 1	**CAP Iteration - 2**	**CAP Iteration - 3**	Conceptual Category
Themes with Common Meanings (Similar)			
	Development Progress	Development Progress	Development Progress
Clarity of requirements	Product Requirements	Product Requirements	Product Requirements
Development	Development	Development Planning	Development
Integration	Integration	Integration	Integration
Whole system view	System View		System View
Themes without Common Meanings (Different)			
Development environment			
User aspects			

Analysis - EAP

Following a similar coding process in EAP iterations resulted in identifying many themes in respective EAP iterations. As shown in Table 3, comparatively the emerged themes were large in number in each of the EAP iteration.

Table 3. Identified themes in EAP iteration 1, 2, and 3.

EAP Iteration - 1	EAP Iteration - 2	EAP Iteration - 3
Development Environment	Development Progress	Development Progress
Highlighted Requirements	Work Done for Future Iterations	Actual Working
Development Planning	Design	User Experience Aspects
Design	Development	Final Review
Product Requirements	Development Delays	Requirements
Requirements Quality	Actual Working	
Collaboration		
Actual Working		
Team Agreement		
Design Approach		
Experience		
Users		

A similar analysis approach identified that group categories were also slightly higher in number than in CAP iterations. Table 4 shows the grouped themes and the formation of respective EAP conceptual categories.

Table 4. Conceptual categories - EAP iteration 1, 2, and 3.

EAP Iteration - 1	**EAP Iteration - 2**	**EAP Iteration - 3**	Conceptual Category
Themes with Common Meanings (Similar)			
	Development Progress	Development Progress	Development Progress
Development Planning		Development	Development
		Development Delays	
Design	Design		Design
Design Approach			
Experience	User Experience Aspects		User Experience
Users			
Highlighted Requirements		Requirements	Requirements
Product Requirements			
Requirements Quality			
Actual Working	Actual Working	Actual Working	Actual Working
Themes with Common Meanings (Different)			
Collaboration	Work Done for Future Iterations	Final Review	
Team Agreement			

As shown in Table 4, there were six 'similar' conceptual categories and four 'different' themes that did not share a conceptual commonality with other themes. Further analysis of conceptual categories in Table 2 and 4 clearly shows the presence of a conceptual category 'user experience' in Table 4 whereas such a category is totally absent in Table 2. This clearly shows the evidence of user experience perspective in EAP iterations.

CONCLUSIONS

In this paper, we have presented the qualitative data analysis of a comparative research study of two ASD development projects based on conceptually different two design approaches.

The main contributions of the paper are of three kinds. First, the evidence as to the incorporation of user experience aspects, little design up-front activities and active engagement of user experience designer in an agile project made a positive user experience perception. Second, the research design presented in this study consisted of four types of evaluation approaches namely: observation data comparison, debrief data comparison, individual system evaluation and comparative system evaluation. Such a research design with multi-analysis approaches can be used in similar research and is a significant contribution to knowledge. Third, the research study clearly showed a framework that can be used to integrate user experience perspective in agile software development.

REFERENCES

Adikari, S. McDonald, and C. Campbell, J. 2009. "Little Design Up-Front: A Design Science Approach to Integrating Usability into Agile Requirements Engineering.", Human-Computer Interaction. New Trends. J. Jacko. Berlin, Heidelberg, Springer**:** pp 549-558.

Adikari, S. McDonald, C. and Campbell, J. 2011. "Voice of user : usability in agile software development.", SETE 2011, Systems Engineering and Test and Evaluation in the Next Decade. Canberra, Australia, SESA.

Adikari, S. McDonald, C. and Campbell, J. 2013. "Reframed Contexts: Design Thinking for Agile User Experience Design.", DUXU/HCII 2013, Berlin, Heidelberg, Springer. pp 3-12.

Adkins, L. 2010. "Coaching Agile Teams: A Companion for ScrumMasters.", Agile Coaches, and Project Managers in Transition. Boston, Addison-Wesley.

Bayer, H., Holtzblatt, K., and Baker, L. 2004. "An Agile Customer-Centered Method: Rapid Contextual Design.", Extreme Programming and Agile Methods - XP/Agile Universe 2004, pp 50-59.

Bevan, N. 2013. "Tutorial - Usability and UX: An integrated approach to design and evaluation.", HCII 2013.

Beyer, H. 2010. "User-Centered Agile Methods.", Morgan & Claypool Publishers.

Boehm, B. W. 1981. "Software Engineering Economics.", Englewood Cliffs, NJ, USA, Prentice Hall.

Broschinsky, D. and Baker, L. 2008. "Using persona with XP at LANDesk Software, an Avocent company", Agile '08, Toronto, Ontario, Canada, IEEE.

Budwig, M., Jeong, S. and Kelkar, K. 2009. "When user experience met agile: A case study.", CHI '09, ACM.

Chamberlain, S., Sharp H., and Maiden, N.A.M. 2006. "Towards a framework for integrating Agile development and usercentred design.", 7th International Conference on Extreme Programming and Agile Processes in Software Engineering, pp 143–153.

Ferreira, J., Noble, J., and Biddle, R. 2007. "Up-Front Interaction Design in Agile Development.", XP 2007, pp 9-16.

Ferreira, J., Sharp, H., and Robinson, H. 2012. "Agile development and user experience design integration as an ongoing achievement in practice.", Agile 2012, pp 11-20.

Fox, D., Sillito, J., and Maurer, F. 2008. "Agile methods and usercentered design: How these two methodologies are being successfully integrated in industry." Agile '08, IEEE, pp 63-72.

Haikara, H. 2007. "Usability in agile software development: extending the interaction design process with personas approach.". XP'07, Springer**,** pp 153-156.

Hevner, A. and Chatterjee, S. 2010. "Design science research in information systems - Theory and Practice.", Springer.

Hevner, A. 2007. "The Three Cycle View of Design Science Research.”, Scandinavian Journal of Information Systems (19:2), pp 87-92.

Hevner, A., March, S.T., Park, J. and Ram, S. 2004. "Design science in information systems research.”, MIS Quarterly (28:1), pp 75-105.

Hiroyuki, M. 2013. “Reconsidering the Notion of User Experience for Human-Centered Design.” HIMI/HCII 2013, Springer, 329-337.

Hussain, Z., Slany, W., and Holzinger, A. 2009. "Current State of Agile User-Centered Design: A Survey.”, HCI and Usability for e-Inclusion, pp 416-427.

Iivari, J. 2007. "A paradigmatic analysis of information systems as a design science.", Scandinavian Journal of Information Systems (19:2), pp 39-64.

ISO (2010). "ISO 9241-210:2010: Human-centred processes for interactive systems.", ISO.

Kautz, K. 2009. “Customer and User Involvement in Agile Software Development.”, LNCS Agile Processes in Software Engineering and Extreme Programming, Springer, pp 168-173.

Kuechler, B. and Vaishnavi, V. 2008. "The emergence of design research in information systems in North America." Journal of Design Research (7:1), pp 1-16.

Lee, J. C. and McCrickard, S. 2007. “Towards extreme(ly) usable software: Exploring tensions between usability and Agile software development.”, Agile ’07, IEEE, pp 59-71.

March, S. T. and Smith, G.F. 1995. "Design and Natural Science Research on Information Technology.", Decision Support Systems (15:4), pp 251-266.

McInerney, P. and Maurer, F. 2005. “UCD in agile projects: dream team or odd couple?.”, Interaction, ACM, pp 19–23.

Wolkerstorfer, P., Tscheligi, M., Sefelin, R., Milchrahm, H., Hussain, Z., Lechner, M, and Shahzad, S. 2008. “Probing an Agile usability process”, CHI ’08 extended abstracts on Human factors in computing systems. ACM, pp 2151–2158.

Rogers, Y., Sharp, H., and Preece, J. 2011. “Interaction design: beyond human-computer interaction.”, John Wiley & Sons.

Roto, V., Law, E., Vermeeren, A., and Hoonhout, J. 2011. “User Experience White Paper - Bringing clarity to the concept of user experience.” Retrieved 02 August 2013, from http://www.allaboutux.org/files/UX-WhitePaper.pdf.

Silva, T. S., Martin, A., Maurer, F., and Silveira, M. 2011. “User-centered design and agile methods: A systematic review.” AGILE 2011. Salt Lake City, UT, USA, IEEE, pp 77 - 86.

Simon, H. A. 1996. “Sciences of the Artificial.” Cambridge, MA, MIT Press.

Slimick, J. 2007. "Tutorial on extreme programming (XP)." Journal of Computing Sciences in Colleges, (23:1), pp 28-28.

Sy, D. and Miller, L. 2008. “Optimizing Agile user-centered design.” CHI'08 Extended Abstracts, Florence, Italy, ACM, pp 3897-3900.

Venable, J. R. and Baskerville, R. 2012. "Eating Our Own Cooking: Toward a More Rigorous Design Science of Research Methods." Electronic Journal of Business Research Methods 10, pp 141-153.

West, D. and Grant, T. 2010. “Agile Development: Mainstream Adoption has Changed Agility.”, Cambridge, MA, Forrester Research.

COPYRIGHT