Maturity Models in the Context of Integrating Agile Development Processes and User Centred Design

Dina Salah El Din Nasr Mostafa

Doctor of Philosophy

University of York
Computer Science

June 2013

Contents

Abstract

Integrating Agile development processes and User Centred Design (UCD) gained increased interest arguably due to three reasons: first, the reported advantages of UCD in enabling developers to understand the users needs, and how the software can support their goals. Second, the deficiency of Agile methods in: providing guidance for developing usable software; and discussing usability, user requirements elicitation and usability evaluation methods. Third, differences between Agile methods and UCD that can make their integration challenging. Agile and User Centred Design Integration (AUCDI) research did not exploit Usability Maturity Models (UMMs) potential. Organizations aspiring to achieve AUCDI can utilize UMMs in assessing its UCD capability, identifying organization's UCD weaknesses and strengths, and planning for improvement. This thesis aims to investigate the suitability of utilizing UMMs in the context of Agile processes. In order to achieve this aim, this thesis conducted: first, a Systematic Literature Review (SLR) that identified and classified AUCDI challenges and explored the proposed practices to deal with them. Second, an interview study of industrial AUCDI attempts that identified the AUCDI difficulties and integration methods. Third, an interview study that evaluated the suitability of two UMMs; Nielsen model and Usability Maturity Model-Human Centrdness Scale (UMM-HCS) for utilization in assessing usability maturity levels in the context of Agile projects. The results of these studies revealed that both models are deficient in their theoretical maturation foundations and scoring scheme and in addressing the activities and challenges associated with AUCDI. As a result of these deficiencies we utilized the results of the SLR and the empirical studies in developing a set of dimensions that represent fundamental elements that affect the AUCDI process. These dimensions were utilized in the development of a descriptive AUCDI Maturity Model that addresses the activities, success factors, and challenges identified within the AUCDI domain.

List of Figures

List of Tables

Acknowledgment

I would like to thank everyone who has provided me with support and advice during my PhD journey. In particular, I would like to thank my supervisor, Professor Richard Paige for his knowledge, support and invaluable help throughout my doctoral studies. I would also like to thank Dr. Paul Cairns for his valuable insights into HCI and for his valuable guidance and support throughout this endeavour.

I would also like to thank my departmental assessor, Professor Tim Kelly whose remarkable attention to detail and sound advice has greatly improved the quality of this thesis.

Special thanks to my external examiner, Professor Helen Sharp for her feedback and suggestions that led to considerable improvement of the thesis.

My deepest gratitude goes to the Agile and user centred design integration experts, Mona Singh, Jim Ungar and Jason Lee Chong who took part in evaluating the Agile and user centred design integration maturity model.

Special thanks go to my parents, and my husband Dr.Drar Abd Al Moneim for their love and patience.

My appreciation goes to my friends Nisreen Makled, Rasha Salah and Inas Ezz for their support.

I am also grateful to the anonymous reviewers of my publications whose comments helped shape my research.

Finally, I gratefully acknowledge the Egyptian government for funding my PhD.

And above all thanks to Allah who has helped me finish this endeavour through all the hardships I have been through.

Declaration

I declare the work in this thesis is solely my own, and was carried out at the University of York. This work has not previously been presented for any other award at this or any other institute. All sources are acknowledged as References. Some of the material presented within this thesis has already been published or reviewed for publication, as listed below:

Conference Papers

1. Dina Salah, Richard Paige. Patterns for Integrating Agile Development Processes and User centred Design. The International Conference on Pattern Languages of Programs (PLoP 2014), Monticello, IL., USA, September 2014.
2. Dina Salah, Richard Paige, Paul Cairns. A Practitioner Perspective on Integrating Agile and User Centered Design. The 28th British HCI Conference (HCI 2014), Southport, UK, September 2014.
3. Dina Salah, Richard Paige, Paul Cairns. A Systematic Literature Review for Agile Development Processes and User Centred Design Integration. 18th International Conference on Evaluation and Assessment in Software Engineering (EASE 2014), London, UK, May 2014.
4. Dina Salah, Richard Paige, Paul Cairns. Towards a Framework for Agile User Centered Design Integration. The 26th British HCI Conference (HCI 2012), Birmingham, UK, September 2012.
5. Dina Salah. A Framework for the Integration of User Centered Design and Agile Software Development Processes. ICSE 2011, Doctoral Symposium: 1132-1133, Hawaii, USA, May 2011.

6. Dina Salah, Richard Paige and Helen Petrie. Practices in the Integration of Agile and UCD Processes - A Reality Check. Agile 2010 - UX Session, Nashville, Tennessee, USA, August 2010.
7. Dina Salah, Helen Petrie and Richard Paige. Towards a Framework for Bridging User-Centered Design and Agile Software Development Processes. 3rd Irish HCI Conference 2009, Dublin, Ireland, September 2009.

Posters

Dina Salah and Richard Paige. A Framework for the Integration of Agile Software Development Processes and User Centered Design (AUCDI). 6th International Symposium on Empirical Software Engineering and Measurement, Lund, Sweden, September 2012.

Chapter 1

Introduction

Agile methods are lightweight methods that claim to tackle limitations of *Plan-Driven Methods* via a compromise between absence of a process and excessive process [88, 105]. *"Agile"* methods embodies a set of lightweight methods or processes sharing core values and principles related to a software development process [23]. Agile processes aim to deal with volatile requirements via discarding upfront, precisely defined plans. They are iterative and are used to develop software incrementally. Agile processes claim to focus on customer involvement, targeted development and documentation efforts and production of working code. They also use self-organising teams to enhance communication methods in order to achieve rapid decision making and cope with volatile requirements. Different Agile processes implement these ideas in different ways. All Agile processes share common values and principles, defined in the *Agile Manifesto* [14], the cornerstone for understanding the essence of Agile development [23].

User Centred Design (UCD) is a set of techniques, methods, procedures and processes as well as a philosophy that places the user at the centre of the development process in a meaningful, appropriate and rigourous ways [39, 59, 95, 219, 241]. The goal of applying UCD is to attempt to satisfy users via producing usable and understandable products that meet their needs and interests [59, 219, 241] in addition to their goals, context of use, abilities and limitations [23].

The integration of Agile methods and UCD is of growing interest (for reasons we detail shortly). We call the process of combining Agile methods and UCD "Agile User Centred Design Integration (AUCDI)".

1.1 Motivation

AUCDI is of growing interest to both researchers and practitioners, arguably due to three reasons:

- **UCD advantages**

  The desire to obtain the reported advantages of UCD on the developed software. UCD reportedly enables developers to understand the needs of the potential users of their software, and how their goals and activities can be best supported by the software thus leading to improved usability and user satisfaction.

- **UCD status in the Agile development processes and the Agile community**

  Agile methods regularly refer to and engage with customers. But customers are not synonymous with users. The Agile community hardly discusses users or user interfaces, thus implying either a negligence of user experience or focus on less sophisticated user experience projects [9]. In particular, none of the major Agile processes explicitly include guidance for how to develop usable software [178]. In addition, the interaction design role, usability, and user interface design in an Agile team is unclear and largely overlooked [20, 45]. Furthermore, principles and practices for understanding and eliciting usability and user requirements and evaluating Agile systems for usability and user experience are generally considerably deficient [156, 178, 267].

- **Differences between Agile development processes and UCD**

  There appear to be philosophical and principled differences between Agile methods and UCD that suggest that their integration will be fundamentally challenging. These differences are embodied in focus, documentation, and evaluation methods as it will be discussed in detail in chapter 3, section 3.2.7.

Improving the effectiveness of UCD in software development is a considerable challenge in many organisations. In the 1990s a number of Usability Maturity Models (UMMs) emerged that aimed to assess the organisation's UCD capability and/or performance. Usability capability is defined as

> A characteristic of a development organisation that determines its ability to consistently develop products with high and com-

petitive level of usability [153].

UCD capability assessment measures the extent of consistently and systematically implementing UCD in the different organisational projects, while UCD performance assessment focuses on the extent of effective implementation of UCD in development projects. These UMMs aim to assist organisations in conducting a systematic current state analysis that evaluates the organisation's ability to consistently develop products with competitive and high usability level via assessing the organisation's strengths and weaknesses in regards to UCD related aspects and accordingly plan for improvement actions [153].

Although Agile and User Centred Design Integration (AUCDI) research is growing, it has not yet exploited the potential of usability maturity models, which is strongly relevant to AUCDI practice. Usability maturity models can be utilised by organisations aspiring to integrate Agile and UCD in order to assess its UCD capability and/or performance and identify its weaknesses and strengths related to UCD and subsequently plan for improvement. Moreover, the process of AUCDI maturation and its constituents has not been directly approached. There is an absence of an AUCDI maturity model that can allow organisations to conduct an analysis of the current state in order to: pinpoint its weaknesses and strengths in deploying Agile processes and UCD, determine whether the organisation is sufficiently mature for AUCDI, and identify the potential challenges that could develop during the AUCDI process in order to mitigate them beforehand.

Based on this motivation, this thesis aims to investigate the suitability of utilising usability maturity models in the context of Agile development processes in order to assess the organisation's UCD capability. This investigation will identify whether the publically available usability maturity models in the public domain address the specifics, activities, success factors and challenges identified within the AUCDI domain.

The observations, findings and lessons learned from the usability maturity models' investigations will be utilised in developing a lightweight descriptive maturity model for integrating Agile development processes and UCD. This can provide organisations with a set of dimensions (fundamental elements that affect the integration of Agile development processes and UCD), processes, and practices that act as a road map for successful AUCDI. The AUCDI maturity model can provide a diagnostic tool to assess an organisation's capability to integrate Agile development processes and UCD.

This model will provide organisations with a thorough understanding of AUCDI specifics, activities, success factors and challenges. The results of this assessment can be communicated to three parties: management, developers and UCD practitioners. First, assessment results can be communicated to management to provide them a with better understanding of the issues involved in integrating Agile development processes and UCD as well as pinpoint AUCDI hindrance. Second, developers can use assessment results to gain a better understanding for usability and UCD. Third, UCD practitioners can use assessment results in pinpointing areas of improvement in usability processes and practices. Further details on the AUCDI maturity model are provided in chapters 8 and 9.

1.2 Research Objectives

As a result of this motivation, a number of research objectives were defined for this thesis:

1. To conduct a A Systematic Literature Review (SLR) to investigate Agile and UCD integration challenges, practices and success factors.
2. To conduct an empirical study to investigate industrial AUCDI attempts, which can be used to verify and complement systematic literature review findings.
3. To investigate the suitability of Usability Maturity Models for assessing usability maturity *in the context* of Agile development processes.
4. To Investigate the existence of a relationship between the success of AUCDI attempts and usability maturity levels.
5. To develop a lightweight, descriptive maturity model for integrating Agile development processes and UCD.

These objectives will be addressed in the main chapters of this thesis, particularly chapters 3, 4, 5, 6, 7, 8, and 9.

1.3 Research Questions

In attempting to address these objectives, the thesis answers a number of research questions (RQ).

1. **RQ1:** What are the challenges that could develop during AUCDI adoption process? (Chapters 3 and 4)
2. **RQ2:** What are the potential success factors for AUCDI? (Chapters 3 and 4)
3. **RQ3:** What are the potential practices for AUCDI? (Chapters 3 and 4)
4. **RQ4:** Is there a method to conduct a current state analysis of organisational and project aspects affecting UCD capability and performance? (Chapters 5, 6 and 7)
5. **RQ5:** Is there a relationship between the success of AUCDI attempts and usability maturity level? (Chapters 6 and 7)
6. **RQ6:** Are the available Usability Maturity Models suitable for utilisation in assessing usability maturity levels in the context of Agile projects? (Chapters 6 and 7)
7. **RQ7:** What is the suitable usability maturity model to assess usability maturity levels in the context of Agile projects? (Chapters 8 and 9)

As a result of answering these questions, the thesis delivers a state of art systematic literature review, results of an interview study of industrial AUCDI attempts, results of an empirical study that utilised two usability maturity models in five AUCDI case studies and a descriptive lightweight maturity model for integrating Agile development processes and UCD.

1.4 Research Method

A Systematic Literature Review was conducted to answer RQ1, RQ2 and RQ3. This SLR investigated AUCDI challenges, strategies and success factors and included a total of 71 AUCDI experience reports, lessons learned, and success and failure AUCDI case studies. This was followed by an interview study that involved 14 participants from 11 companies in five different countries to investigate current industrial practices for integrating Agile development processes and UCD to investigate the same three research questions. This empirical study investigated the difficulties and concerns that hinder AUCDI attempts, the integration methods, and compared our findings to the findings of other researchers who investigated the AUCDI attempts.

Then a literature review for usability maturity models was conducted to answer RQ4. This resulted in choosing two usability maturity models, Nielsen Corporate Usability Maturity Model [215] and Usability Maturity Model - Human Centredness Scale (UMM-HCS) [68]. In order to answer RQ5 and RQ6 both Nielsen Corporate Usability Maturity Model [215] and UMM-HCS [68] were applied on five AUCDI case studies via one to one semi structured interviews. The purpose of this interview study was two fold: first, to investigate the relationship between the success of AUCDI attempts and usability maturity level. Second, to investigate the suitability of usability maturity models for utilisation in the context of Agile development processes.

The results, lessons learned and observations from applying Nielsen Corporate Usability Maturity Model [215] and UMM-HCS [68] on five AUCDI case studies as well as the results of the SLR and the empirical study for 12 industrial attempts were utilised in answering the final research question (RQ7).

1.5 Research Contributions

The thesis contributes to the body of research from a variety of perspectives including a systematic literature review and a set of empirical studies. Further details on this research contributions are provided in the following sections.

1.5.1 AUCDI Systematic Literature Review

The first research contribution is presented in chapter 3, a state of art SLR that identifies and classifies various challenging factors that restrict AUCDI and explores the proposed practices and success factors to deal with these challenges. The SLR included a total of 71 papers and excluded 80 papers that were published from the year 2000 till 2012. The SLR findings were quantitatively and qualitatively classified and a description and taxonomy of AUCDI challenges and its respective success factors and practices were reported.

This SLR is considered as a contribution to both academic researchers and industrial practitioners. Industrial practitioners can utilise the description and taxonomy of AUCDI challenges and corresponding practices and success factors in identifying potential challenges of AUCDI and practices or success factors to deal with these anticipated challenges. Academic researchers and industrial practitioners can benefit from this SLR in identifying research areas that exhibit apparent scarcity and thus need to be further exploited via future research work.

1.5.2 Conducting an Empirical Study of Industrial Practices for AUCDI

The second research contribution is presented in chapter 4 , an empirical study that investigated industrial AUCDI attempts in order to verify and complement literature findings. A set of interviews were conducted and analysed that involved 14 participants from 11 companies in five different countries: United Kingdom, Canada, Netherlands, Portugal, and Egypt. These interviews identified the difficulties that hinder AUCDI attempts, the integration methods used, and confirmed and complemented findings of other researchers that were revealed by the SLR.

1.5.3 Investigating Usability Maturity Models Role in AUCDI

The third research contribution is presented in chapters 6 and 7. These chapters reported on five one to one semi structured interviews that evaluated the suitability of UMMs for utilisation in assessing usability maturity levels in the context of Agile projects and investigated the existence of a relationship between the success of AUCDI attempts and usability maturity levels. This occurred via utilising two UMMs: Nielsen Model and Usability Maturity Model-Human Centrdness Scale (UMM-HCS) in five AUCDI case studies and assessing their usability maturity levels.

Findings, observations and lessons learned from chapters 6 and 7 revealed that both models are deficient in their theoretical foundations with respect to maturation, scoring scheme, advice on the assessment of criteria, terminology used and accuracy of some of the key practices.

The investigation of the existence of a relationship between the success of AUCDI attempts and usability maturity level via Nielsen model gave an indication of the existence of a correlation between the success of AUCDI attempts and the usability maturity level of the AUCDI case study. The investigation via UMM-HCS revealed that it was not possible to achieve this aim due to the model's linear upgrading which led to discarding considerable achieved attributes by the five case studies. The results of the five case studies gave an indication that the linear model of upgrading is contradictory to how organisation's perform since an organisation can score high in some of the practices related to a high usability maturity level even if the organisation is at a low usability maturity level.

Although both Nielsen model and UMM-HCS do not conflict with Agile values and principles yet both models do not address the specifics, activities, success factors and challenges identified within the AUCDI domain and subsequently they do not pinpoint all dimensions and practices involved in the AUCDI process. Both models do not have details on the timing and the lightweight method of applying the different UCD practices along the Agile development life cycle iterations or sprints. These issues need to be taken into consideration by any researcher who considers to develop a usability maturity model for utilisation in the context of Agile development processes. Examples of AUCDI challenges that are not approached by both models are: practices regarding the communication, coordination and collaboration between UCD practitioners and Agile developers in order to synchronize and complete their work, practices related to design modularization and chunking, and maintaining commu-

nication between the customer and the development team. Another issue that needs to be approached by the developers of usability maturity model for utilisation in the context of Agile development processes is the features and activities of some team roles that are introduced by Agile methods. These team roles are: XP coach, Scrum master, product manager whose role can impact the integration process.

1.5.4 Developing a Set of Dimensions for AUCDI

The fourth research contribution is presented in chapter 8. The results of the studies described in chapters 3, 4 , 6 and 7 contributed to the development of a set of dimensions for integrating Agile development processes and UCD. AUCDI is dependent on four main dimensions: UCD infrastructure that lays the necessary foundation for the integration; AUCDI process that includes detailed activities, work products, work flows, roles, and responsibilities; people involved in the integration process; and UCD continuous improvement.

1.5.5 Development of an AUCDI Maturity Model

The fifth research contribution is presented in chapter 9, a descriptive AUCDI Maturity Model is developed. The maturity model addresses the specifics, activities, success factors and challenges identified within the AUCDI domain. The AUCDI Maturity Model can be used by organisations for two purposes: AUCDI process definition and assessment.

The AUCDI maturity model is up to our knowledge the first maturity model to approach the process of AUCDI maturation and its constituents. The maturity model is composed of three components: first, a multidimensional reference model that includes a set of fundamental elements that affect AUCDI and thus should be reflected in the model and examined in an assessment. These dimensions are: UCD infrastructure; AUCDI process; people; and UCD continuous improvement. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. Third, an assessment procedure to provide practical guidance for performing the assessment. The assessment procedure is composed of a performance scale, maturity recording sheet, maturity levels, typical quotes, and assessment guidelines.

1.6 Chapter Summary

This chapter provided an overview of the thesis. It presented the motivations, research objectives, scope, research questions, research methodology and main contributions of the thesis. The structure of the remaining chapters has also been delineated.

The next chapter will discuss the important foundational concepts relevant to the thesis. These concepts are the Agile Manifesto, Agile processes, user centred design and Agile and UCD integration and maturity models.

Throughout this thesis the term UCD practitioner will be used as a synonym for a professional, regardless of his or her education or job description, who is experienced in usability engineering and is responsible for good user interface design, practicing of usability methods, and the implementation of UCD in the organisation [103]. This term is used instead of the following terms: UCD specialist, usability specialist, interface designer, designer, interaction designer, usability engineer, user experience designer, etc.

Chapter 2

Background

This chapter provides a foundation for important concepts relevant to the thesis. This foundation is essential for comprehending the thesis motivation and basics. It starts by introducing the Agile Manifesto, emphasizing Agile values and principles. Then it focuses on two examples for Agile processes: Extreme Programming and Scrum. This is followed by defining interaction design, usability and user experience, and discussing the definition, basic principles and activities of UCD. The final part of this chapter focuses on maturity models and discusses maturity models' definition, examples, purposes, critique, and design. Then the chapter ends by discussing usability maturity models as related to subsequent chapters.

2.1 The Agile Manifesto

A Software Development Methodology (SDM) is defined as

> A recommended collection of phases, procedures, rules, techniques, tools, documentation, management and training used to develop a system [10].

Plan Driven Methodologies are often referred to as heavyweight methodologies [105]. This is attributed to the fact that they are process centric, utilise a predefined plan for guiding projects' life cycle and follow a repeatable and optimized process [88, 105]. Examples of plan driven methodologies are: the Personal Software Process (PSP) [121], the Team Software Process (TSP) [122], and the Rational Unified Process (RUP) [169]. These methodologies often produce considerable documentation that codifies software and process knowledge. The bureaucracy inherent in these methodologies led to the evolution of lightweight methodologies [88, 105].

The main differences between heavyweight and lightweight methodologies are that the latter are adaptive rather than predictive since heavyweight methods are change resistant and tend to considerably detail the software process while lightweight methods welcome changes and adaptability [88]. Moreover, lightweight methods are people oriented rather than process oriented [88].

Agile methods are lightweight methods that claim to tackle perceived limitations of *Plan-Driven Methods* via a compromise between absence of a process and excessive process [88, 105]. *"Agile"* embodies a set of lightweight methods or processes that share core values and principles related to a software development process [23]. Barnett [12] conducted a survey that required participants to rank ten different reasons for adopting Agile methods and to estimate specific quantitative levels of improvement (i.e., none, 10%, 25%, or greater than 25%) in four areas. The responses indicated that Agile development has led to an increase in 1700 companies' success rate. This success was in the form of a number of advantages as a result of turning to Agile: increased productivity (83% reported 10% or higher enhancements), quicker time to market, decreased cost (28% of respondents reported 25% or higher improvements), reduced software defects (54% of respondents reported 25% or higher enhancements). Other reported improvements in dealing with changed priorities and enhanced team moral and [12].

Agile processes aim to deal with volatile requirements via discarding upfront, precisely defined plans. They are iterative and are used to develop software incrementally. Agile processes claim to focus on customer involvement, targeted development and documentation efforts and production of working code. They also use self-organising teams to enhance communication methods in order to achieve rapid decision making and cope with volatile requirements. Different Agile processes implement these ideas in different ways. All Agile processes share common values and principles, defined in the *Agile Manifesto* [14], the cornerstone for understanding the essence of Agile development [23]. The Agile Manifesto resulted from a meeting that was held in February 2001 between 17 representatives of the different Agile methods to discuss and promote their views on lightweight methodologies. This meeting resulted in the emergence of the Agile Manifesto and the formation of the Agile Alliance. The Manifesto's purpose is to reveal the essence of Agile processes, provide a common foundation for the different Agile methods and emphasize the values and principles embodied in all Agile methods.

2.1.1 Agile Values

Agile Values are regarded as relative statements used for alternatives weighing rather than absolutes [24]. Agile values [14] are as follows:

- **Individuals and Interactions Over Processes and Tools**

  People rather than processes and tools are perceived as the key factor in developing software by Agile methods as a result Agile methods values individual skills and depend on self-organising, proactive, and cohesive teams that effectively utilise face to face communication in daily work interactions [23]. This focus on individuals and interactions contrasts with plan driven methodologies focus on institutionalized development tools and processes.

- **Working Software Over Comprehensive Documentation**

  Agile teams main focus is to frequently deliver tested working software with minimal sufficient documentation rather than producing up to date and complete documentation.

- **Customer Collaboration Over Contract Negotiation**

  Agile methods strive to achieve customer satisfaction via close in-

teraction rather than depending solely upon contracts. It is recommended for customers to be collocated with development teams to achieve close daily interaction. However, contract negotiation is still acknowledged as a method to attain and maintain strong customer relationship.

- **Responding to Change Over Following a Plan**

  Agile methods welcomes requirements change and cope with volatile requirements via discarding upfront, precisely defined plans. However, Agile methods focus on providing the development team with the competence and authority to both adjust and generate flexible plans that accommodate change rather than avoid planning altogether.

2.1.2 Agile Principles

Agile values are encoded into a set of more concrete *Agile Principles* defined in the Manifesto. Agile principles are regarded as relative statements used for alternatives weighing rather than dichotomous. These principles are fulfilled in different Agile methods in various ways. It is significantly important to discuss these principles since any attempt for integrating Agile and UCD should pay equal attention to following the principles of Agile as well as following the principles of UCD or else its proposed integration approach can result in the emergence of a new process that is neither Agile nor user centred. The Manifesto describes the following principles [14].

- Customer satisfaction via early and continuous delivery of valuable software.
- Requirements change is welcomed.
- Frequent delivery of working software
- Developers and business people must work daily together all through the project.
- Projects should be formed of motivated team members who are trusted and provided with supportive environment.
- The most effective and efficient technique for conveying information is face to face conversation.
- The main progress measure is working software.

- Sustainable development is promoted and developers, sponsors and users should strive for constant pace.
- Agility is enhanced via continuous focus on good design and technical excellence.
- Simplicity is essential.
- Self-organising teams result in the emergence of the best requirements, architectures, and designs.
- Frequent team reflection on work is essential for continuous adjustment and improvement.

2.2 Agile Methods

The above principles do not prescribe any specific process guidance. The different *Agile Methods* choose to provide their own process guidance that embraces the Manifesto's values and principles. A representative list of Agile methods include but are not limited to *Extreme Programming (XP)* [15], *Scrum* [255, 256, 257], *Adaptive Software Development (ASD)* [112], *Dynamic Systems Development Method (DSDM)* [268, 269], *Agile Modeling (AM)* [6], *Crystal* [42] and *Feature Driven Development(FDD)* [225]. This section focuses specifically on two of the most well known Agile methods: XP and Scrum. The reason behind choosing those two methods is that according to the results of the SLR that was conducted and reported in chapter 3, only two papers among all published literature on Agile and UCD integration focused on Agile methods other than XP and Scrum; Haikara [102] focused was on Mobile-D, while Krohn et al. [168] focused was on FDD. As a result it was decided to provide the essential XP and Scrum concepts that can provide a foundation for understanding subsequent chapters.

2.2.1 Extreme Programming

Extreme programming (XP) [15] is an incremental, iterative approach that utilises skilled teams to satisfy customers via delivering a set of working code releases that satisfy a defined set of customer requirements [15, 171]. XP development process starts by generating a preliminary high level requirements list, then estimating implementation time for list items and

then prioritizing and scheduling their execution for the first system release. Then iteratively developing, deploying and incrementally improving the first release in subsequent iterations [282]. Figure 2.1 embodies the development process for Extreme Programming.

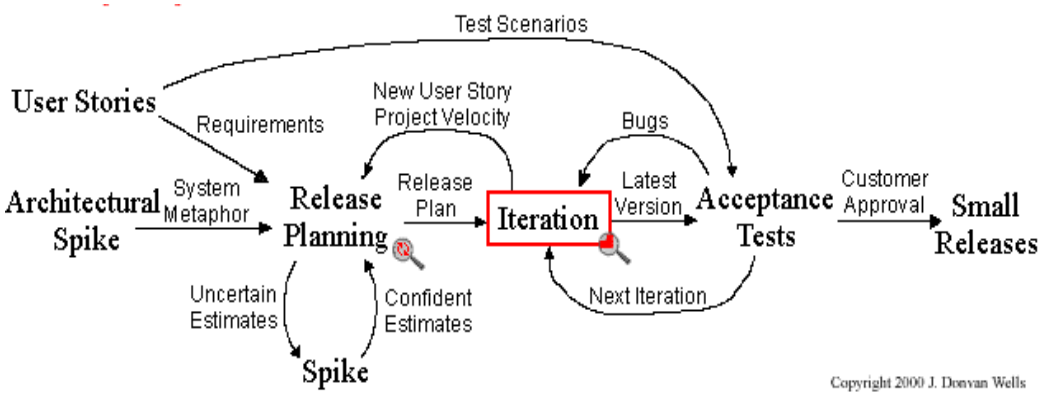

Figure 2.1: XP Development Process [282]

2.2.2 Scrum

Scrum is an incremental and iterative process focused on software project management rather than defining specific practices for developing software. Thus Scrum is considered as a process outline, and a set of technical roles [255, 256, 257].

The Scrum development process starts off by producing an initial list of prioritized system requirements that are stored in the product backlog. This is followed by a sprint planning meeting in which project risk analysis, resources estimate and project scheduling is performed. Then a number of backlog items are chosen to be implemented during the next sprint (typically one month in length). Each Sprint finishes by completing a predetermined subset of the product backlog. Then all those increments are integrated together and released into the user environment [257].

Scrum depends on a number of meetings: the daily Scrum and the sprint retrospective. The daily Scrum meeting identifies what team members have done since the previous day, what they plan to do, and what problems they have identified that may impede progress. In the sprint retrospective, team members attempt to identify what worked well in the

sprint, and what could be improved [257]. Figure 2.2 illustrates the development process of Scrum.

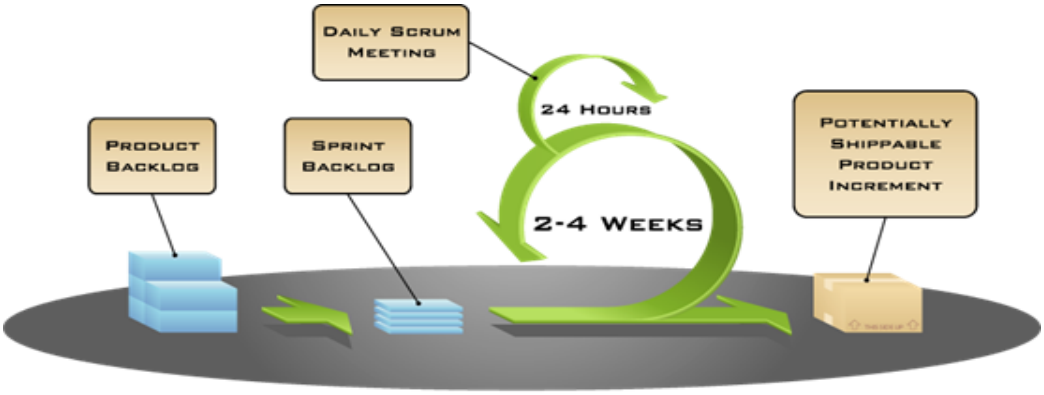

Figure 2.2: Scrum Development Process

Scrum Roles

Scrum defines a set of roles; some of these roles are involved (modified or as-is) in the various integration approaches for Agile and UCD that are discussed in chapter 3. In addition, chapter 8 refers to these roles and their effect on integrating Agile and UCD. These roles are as follows [2]:

- *Scrum Master*

  The *Scrum Master* interacts with the development team, management and customer throughout the project to ensure Scrum values and practices are followed. He also acts as a problem solver and removes any impediments to ensure best team productivity.

- *Product Owner*

  The *Product Owner* is the stakeholders' representative and is responsible for participating in estimating the development effort for backlog items and turning the backlog items into developed features.

- *Team*

  The *Scrum team* is a cross-functional self organising team that sets sprint goals, estimate effort, review product backlog list and communicates any impediments to the Scrum master.

- *Customer*

The *Customer* participates in product backlog tasks for newly developed or enhanced systems.

- *Management*

  The *Management* participates in requirements and goals setting and are in charge of final decision making. For example, they are responsible for selecting the product owner, gauging project progress and determining backlog items with the Scrum master.

2.3 Interaction Design

In order to understand the challenges involved in integrating Agile and UCD. It is first necessary to discuss UCD and its related terminology, for example, interaction design, usability and user experience since all those terms are repeatedly referred to in chapter 3 onwards, thus discussing them lays essential stepping stones for the following chapters.

Interaction design is achieved via a user centred approach to design, it involves four iterative core activities: identifying user needs and requirements, developing alternative designs, building interactive design versions, and evaluating the usability of what is being built throughout the process [241]. The interaction design process has three key characteristics: user involvement throughout the development process, early identification and documentation of user experience and usability goals and iteration through interaction design activities [241].

2.4 Usability

The *Usability* of a product is the consequence of systematic UCD work that occurs throughout the development process and continues even after product release in order to enhance subsequent versions [59, 97]. Usability has been referred to by various definitions that convey different meanings, making it a very perplexing concept, especially for software developers [19]. Usability has not been consistently defined by researchers, standardization organisations or the software development industry [259]. The various definitions of usability will be highlighted below:

The ISO has developed different standards related to usability, and two major categories can be distinguished; product oriented standards (ISO 9126 and ISO 14598) and process oriented standards (ISO 9241 and ISO 13407). Three standards has perceived usability differently, ISO standard 1991, 1996 and IEEE standard 1998, as can be illustrated below

> The capability of the software product to be understood, learned, used and attractive to the user, when used under specific conditions [137] as cited in [259].
>
> The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use [138] as cited in [259].
>
> The ease with which a user can learn to operate, prepare inputs for and interpret outputs of a system or component [134] as cited in [259].

Beside the definitions of usability offered in ISO and IEEE standards, a number of other researchers introduce their own definitions, for example, Jokela [149] define usability as

> A quality attribute of a product that is dependent on the extent and performance of UCD activities in a specific development project [152].

Nielsen and Phillips [216] define usability as

> The absence of obstacles that prevent users from completing their tasks with the system [216].

This definition implies that a high number of identified usability problems usually indicates a low degree of usability [265].

Gould and Lewis [95] declares that any system designed with the intention for people to use should be easy to remember, easy to learn, useful and pleasant to use.

Preece et al. [241], points out that usability ensures optimizations of people interactions with interactive products. Usable products are characterised by being easy to learn, enjoyable and effective to use from the user's perspective. Usability is scrutinized into a set of goals comprising efficiency, effectiveness, utility, learnability, safety, and memorability. Those goals are turned into usability criteria that facilitate product assessment. Examples of usability criteria are time to learn a task (learnability), time to complete

a task (efficiency), and number of errors made when performing a given task over time (memorability) [241].

It can be concluded from the above definitions that there is a lack of a standardized definition for usability which places an additional burden on practitioners. A common factor between all definitions is that usability is related to users, goals, and contexts of use. Seffah and Metzker [259] stated that although the different ISO definitions are conceptually clear yet they make it difficult to specify and interpret measurable usability attributes. Moreover, it is hard to figure out how the three factors (effectiveness, efficiency, and satisfaction) in [138] and their respective indicators contribute to an overall conclusion about the system usability [259].

As for the definition of usability that was brought forward by Jokela and Abrahamsson [152] who suggested that usability is dependent on the extent and performance of UCD activities. This definition did not clarify the specific UCD activities that can help achieve usability. This is confusing to practitioners since there is a variety of proposed UCD principles and activities discussed by a number of researchers and in a number of ISO standards: ISO 13407, [95], [214], [97], and Jokela and Abrahamsson [152] did not clarify the involved UCD activities that was referred to. Also this definition did not specify any measurement criteria for judging the product usability.

In regards to Nielsen and Phillips [216] definition of usability, this definition did not specify any criteria for measuring usability. Gould and Lewis [95] definition specified the criteria for judging product usability yet it did not mention how can this usability be achieved. The final definition by Preece et al. [241] is more comprehensive and clear since it defines the qualities of usable product, usability goals and it translated those goals into a set of measurable criteria that can be easily utilised by practitioners to design for and measure usability.

2.5 User Experience (UX)

User Experience is another concept whose definition is still being explored and debated by academic researchers and industrial practitioners [173].

The (ISO standard 9241-210, 2008) defines UX as:

> A person's perceptions and responses that result from the use

or anticipated use of a product, system or service.

Garrett [92] defines user experience as the experience that users go through via using the product. UX focus is on how a user perceives a product that he works with. UX is perceived as a critical criteria in differentiating between a successful and unsuccessful product [92].

Preece et al. [241] points out that the emerged diversity of application areas (e.g.public areas, education, entertainment, home) and advanced technologies that cater for their needs (e.g., virtual reality, the web, mobile computing) has resulted in a broader set of concerns. In addition to the primary focus of improving productivity and efficiency, interaction design is embracing other goals that aim to focus on UX and develop interactive products that are enjoyable, satisfying, motivating, aesthetically pleasing, entertaining, helpful, fun, rewarding, supportive of creativity, and emotionally fulfilling [241].

As can be perceived from the above definitions that UX is subjective since it is related to how a user evaluates his own experience with using a product. Moreover, UX is not relevant to all product types but only to educational or entertainment products and the similar. Nevertheless, a point worth noting is that some of the AUCDI literature repeatedly refers to UX as an equivalent term to usability, the reason behind that could be that a large portion of the published papers on integrating Agile and UCD is written by Agile practitioners who are not UCD experts. More on lack of UCD skills in Agile teams will be covered in chapter 3.

2.6 User Centred Design

User Centred Design (UCD) is a set of techniques, methods, procedures and processes as well as a philosophy that places the user at the centre of the development process in a meaningful, appropriate and rigourous ways [39, 59, 95, 219, 241]. The goal of applying UCD is to attempt to satisfy users via producing usable and understandable products that meet their needs and interests [59, 219, 241] in addition to their goals, context of use, abilities and limitations [23]. The usability of a product is the consequence of systematic UCD work that occurs throughout the development process and continues even after product release in order to enhance subsequent versions [59, 97].

Nevertheless, technology is the dominant driver to the software develop-

ment process rather than users' goals, needs, context of use, limitations and abilities [23]. However, software teams that avoid customers' input actually receive the input following the product release in the form as lost sales, rewrites, cancelled contracts and bad reviews and results in lost revenue, development time and reputation [203].

Gould and Lewis [95] recommends three principles of UCD. First, early focus on users and tasks, where designers study and understand users' cognitive, attitudinal, anthropometric, and behavioural characteristics and the nature of the tasks to be accomplished by users. Second, empirical measurement, via utilising prototypes to allow users to perform tasks and recording, observing and analysing their reactions and performance. Finally, iterative design is utilised to fix any problems discovered through user testing. Gould and Lewis [95] points out that it is crucial to have a repeated cycle of design, test and measure, and redesign.

Nielsen [214] proposes a model for UCD that has a number of practices: identifying users tasks and roles, competitive analysis, setting usability goals, developing and exploring different design alternatives, user involvement and feedback via participatory design, ensuring consistent and coordinated design of the user interface, applying heuristic analysis and guidelines, early prototyping preferably paper prototypes, empirical usability testing, iterative design then after the release of the product collecting feedback from field use [214].

In addition, UCD processes are defined in a number of ISO standards, for example, ISO 13407 (Human Centred design processes for interactive systems), an international standard established in 1999 and is considered as a guidance and reference model for UCD [153]. Another associated technical report that discussed UCD was the ISO TR 18529. ISO 13407 defines UCD as

> An approach to interactive system development that focuses specifically on making systems usable.

ISO 13407 is focused on explicating a set of method independent UCD principles and activities that assist in creating usable systems or products and allows organisations to understand the UCD status in their development process [150]. ISO 13407 principles are: multi-disciplinary teamwork, active users involvement, iterative design, and appropriate functions allocation between the users and the system as cited in [153]. Figure 2.3 shows UCD activities according to ISO 13407.

The four key UCD activities (processes) defined by ISO 13407 as cited in

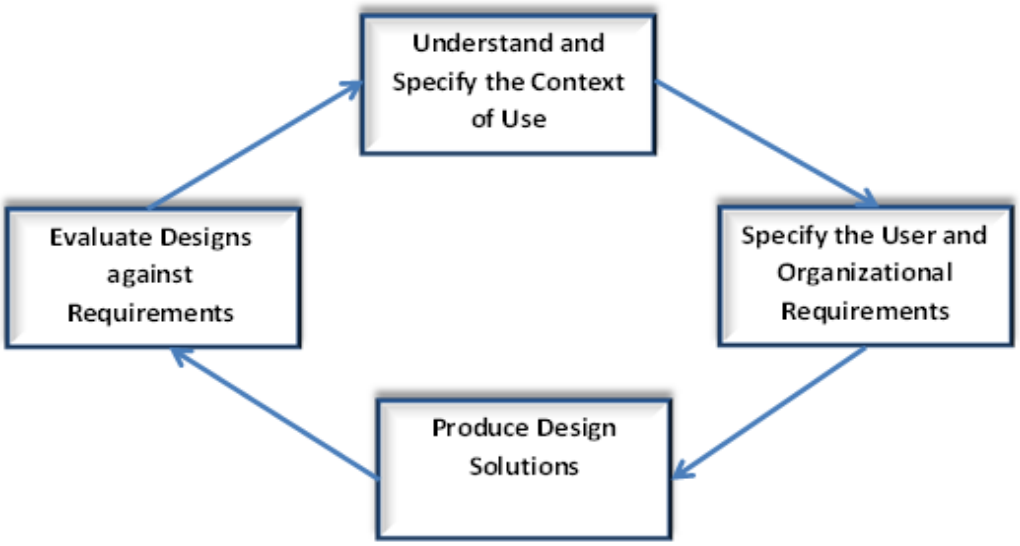

Figure 2.3: User Centred Design Activities According to ISO 13407

[144] are as follows:

- Understand and specify context of use. This activity involves identifying and understanding the user, environment of use, and user tasks.
- Identify the user and organisational requirements. This activity involves pinpointing the success criteria related to the usability of the product in terms of user tasks. It also involves determining the design constraints and guidelines.
- Produce design solutions via incorporating interaction design, visual design and usability related knowledge.
- Evaluate designs against requirements.

Preece et al. [241], discussed user centred development and elaborated on Gould's first principle for UCD design [95] by suggesting five further principles that expand and clarify on Gould's first UCD principle "early focus on users and tasks". These five principles are: user's goals and tasks that are perceived as the development drivers, studying users' behaviour and context of use and designing the system to support them, capturing and designing users' characteristics, early and continuous user consulting throughout the development process and taking their input into account, and influencing and driving by context of use, environment of use and users' tasks [241].

Gulliksen et al. [97] argued that the UCD principles listed in ISO 13047 standards are insufficient to maintain UCD and the UCD process needs to be specified in more details. Thus he proposed 12 key principles for UCD: focus on user context of use, tasks, goals and needs, early and continuous active involvement of users in the development process and system life cycle, iterative and incremental systems development, simple design representations, early and continuous prototyping, evaluating design against usability goals and design criteria in cooperation with end users, including conscious and explicit design activity inside the development process, choosing effective multidisciplinary teams, early and continual involvement of a usability champion throughout the development life cycle, maintaining holistic design, customization of UCD process locally via specifying and adapting it, and finally maintaining a user centred attitude [97].

Jokela [146] proposed KESSU model, a usability assessment model that evolved as a result of lessons learned and concrete interpretations from utilising Usability Maturity Model - Processes (UMM-P) in a number of industrial case studies. KESSU 2.2 proposed a new UCD reference model that took the UCD processes of ISO 13407 and ISO 18529 as its base. KESUU 2.2 characteristic features are the identification of six UCD processes rather than the four processes used in ISO 13407 and ISO 18529, identification of two process categories: user interaction design and usability engineering, and the utilisation of outcomes to define processes. The validation of the model via case studies revealed that these six UCD processes make UCD activities easier to define and understand and focus on the integration of usability engineering and interaction design [146].

KESSU 2.2 processes and its outcomes can be shown in figure 2.4

KESSU 2.2 processes and its outcomes are as follows:

1. **Identification of User Groups**

   This process identifies the potential user groups. Its outcomes are the definition of user groups and user characteristics [144].

2. **Identification of Context of Use**

   This process identifies the characteristics of users and tasks. It also identifies the organisational, technical, and physical environment of the developed product. Its outcomes are user group instances as well as context of use of the new and old system [144].

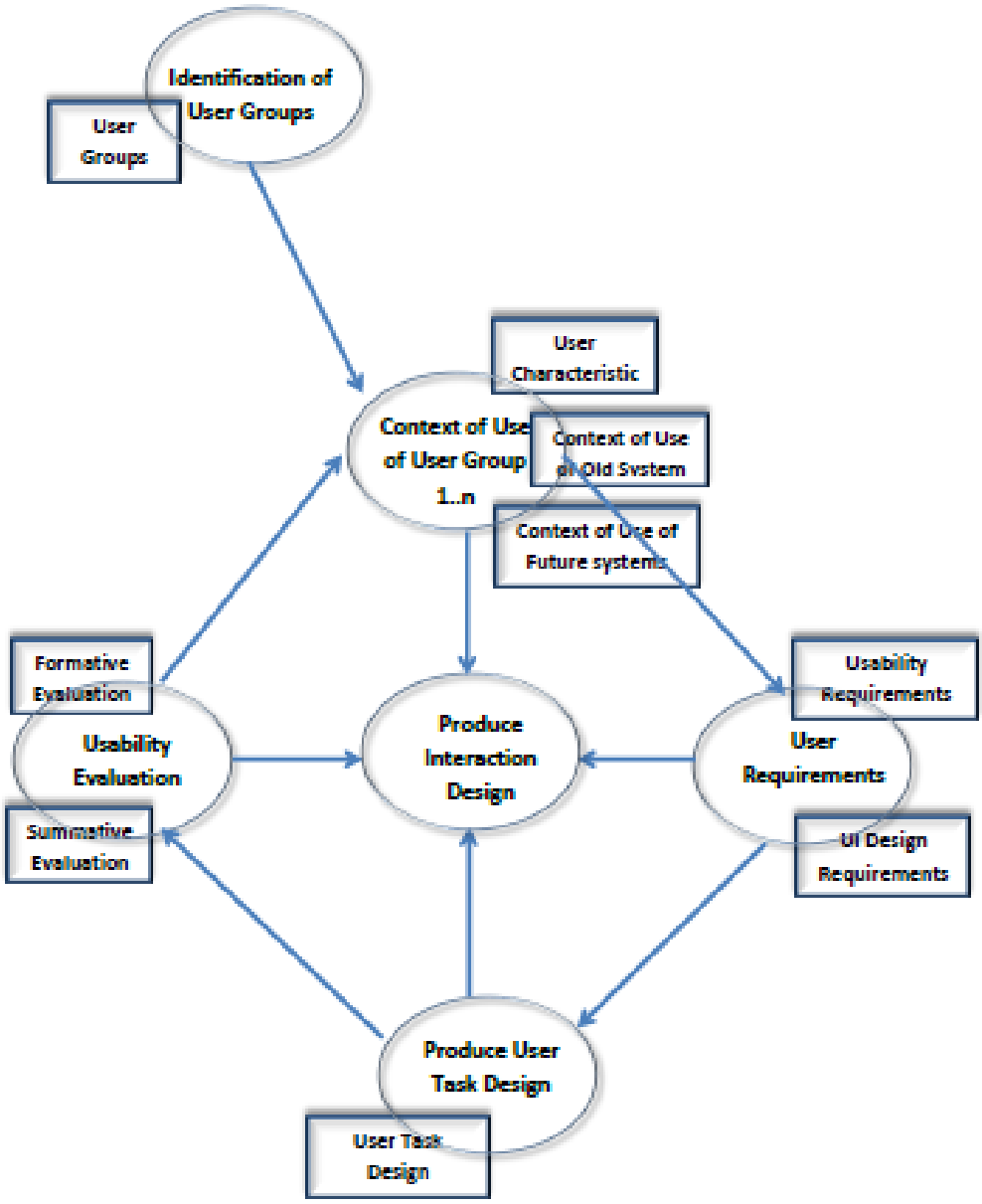

Figure 2.4: KESSU 2.2 UCD Process Model

3. **Identification of User Requirements**

   This process defines the product's usability and user interface design requirements. The context of use information and the business goals of the project are fed into the process. Its outcomes are: first, the determination and the documentation of usability requirements against the context of use. Second, the determination and documentation of the user interface design requirements via style guides, company or project standards or user interface design heuristics [144].

4. **Designing User Tasks**

   This process focuses on designing how users would use the new product to perform tasks. Its outcomes are the description and documentation of user tasks [144].

5. **Designing User Interaction**

   This process focuses on designing product elements that users interact with. Its outcomes are user interface, user documentation, user support procedures and user training [144].

6. **Evaluating Usability**

   This process focuses on the iterative evaluation of the product against the requirements. Its outcomes are summative and formative usability evaluation results. Summative usability evaluation results evaluate the extent to which the product meets its defined usability requirements while formative usability evaluation results collect qualitative usability feedback.

Chapter 8 will discuss UCD activities and principles that were chosen to become the basis of the assessment model for integrating Agile development processes and user centred design.

2.6.1 User Centred Design Frameworks

This section discusses three examples of UCD frameworks: contextual design, scenario based design and usability engineering life cycle. These UCD frameworks are investigated and utilised by both UCD and Agile practitioners to achieve Agile and UCD integration. Those frameworks are repeatedly referred to in chapter 3, that reports the results of a SLR that investigated the challenges, strategies and success factors for AUCDI.

Those frameworks are also referred to in chapter 4 that reports the results of an empirical study for industrial attempts to integrate Agile and UCD.

2.6.1.1 Contextual Design

Contextual Design [20, 21] is a structured approach to gathering and representing fieldwork information in order to utilise it in design endeavours. Contextual design has seven parts: contextual inquiry, work modeling, consolidation, work redesign, user environment design, mock up and test with customers and putting it into practice [241].

Contextual Design process techniques involves the following [20, 21]:

Contextual Inquiry: This involves interviewing users in their work places in order to observe and investigate their daily work activities and practices.

Interpretation Sessions and Work Modeling: Interpretation sessions are team discussions that recaptures the key points of the contextual inquiry. In addition, work models are sketched for users' work practice. Both interpretation sessions and work modeling contributes to a common comprehension of user and project data.

Consolidation and Affinity Building: This involves consolidating users' data and work models in an affinity diagram to show users' issues and common strategies and work patterns across all users.

Visioning: Visioning involves sketching work models and scrutinizing methods by which the system will address the work practice.

Story Boarding: A storyboard is drawn for the to-be work model manual. This work model manual explicates new practices, business rules, initial UI concepts and automation assumptions.

User Environment Design (UED): A coherent design of all system functions is built from the storyboards. This provides a basis for prioritization and system segmentation.

Paper Prototypes and Mock-Up Interviews: Paper prototypes are prepared for user interfaces and tested with the system's actual users to verify the basic system function and UI design.

2.6.1.2 Scenario Based Design

A *Scenario* is "an informal narrative description" [37]. Scenarios describe user tasks and activities in a story form thus allowing the exploration and discussion of requirements, contexts, and needs [241]. *Scenario Based Design* revolves around combining scenarios usage with claims that express the negative and positive effects of certain design features as a foundation for interactive systems creation [38, 250]. Scenarios utilise the vocabulary and phrasing of user thus they are easy to understand by stakeholders and form a powerful mechanism for communicating among team members and with users. Stakeholders are usually actively involved in producing and validating scenarios [241]. Scenario based design specifies involves the following four design phases [38, 250].

Requirements Analysis: involves designers conducting information collection on current practices via ethnographic studies, interviews and other techniques for gathering data. The gathered data is utilised to describe the overall system vision and stakeholder descriptions and build a root concept document. Then designers work on problem scenarios and claims that pinpoint key problems and issues and how current process is performed [38, 250]. **Activity Design:** involves the designers using the developed problem scenarios and claims to describe tasks and activities [38, 250].

Information Design: involves designers using the information provided by the interface and its supported interactions to determine the methods that will be used to support activities. This is usually performed iteratively and then followed by usability evaluation for the design to ensure it satisfies the requirements analysis goals [38, 250].

Interaction Design: involves designers working iteratively on the interaction design that supports user tasks and activities and then usability evaluation is conducted to ensure the interaction design satisfies the requirements analysis goals [38, 250].

2.6.1.3 Usability Engineering

Usability Engineering involves specifying formal, verifiable and quantifiable usability criteria [214]. This is achieved via determining measurable criteria for product performance and assessing the product against these

measures then utilising the assessment results in making changes to subsequent system versions [241].

Mayhew [191] proposed a *Usability Engineering Life Cycle* which provides a holistic view of usability engineering and a detailed description of the proposed methods for executing usability tasks and integrating it into traditional software development life cycles [191]. The usability engineering life cycle has essentially three tasks: requirements analysis, design/testing/development, and installation [241].

Figure 2.5 clarifies the different phases of the usability engineering life cycle.

2.6.2 Requirements Elicitation and Usability Evaluation Techniques

This section focuses on a number of techniques that are used for requirements elicitation in order to collect relevant, sufficient, and appropriate data. These techniques are referred to in subsequent thesis chapters. Those include questionnaires, interviews, focus groups and workshops, and naturalistic observation [241]. Questionnaires, interviews, focus groups and workshops are considered to be inquisitive techniques, nevertheless, they are subjective, however, naturalistic observations provide real time portrayal of the studied phenomena [163].

Questionnaires: *Questionnaires* are a set of questions designed to collect specific information and are usually remotely administered; sent electronically or posted on a website, and sometimes they are delivered on paper [163, 241]. Questionnaires are time and cost effective [163] and can be used to collect data from a large number of participants in geographically diverse locations [163, 241]. Since questionnaires are usually filled by the respondents in the absence of its creator thus considerable thought and preparation should be given to the questions' wording, layout, and ordering in order to ensure valid results [163, 241]. Questionnaires also suffer from relatively low response rates which can affect the sample representativeness and the population homogeneity. The sampling technique used also affects the extent of the results generalization [163].

Interviews: *Interviews* involve asking someone a series of questions and tend to be one to one in order to elicit one person's perspective at a time [163, 241]. Interviews can be classified into structured, semi structured or

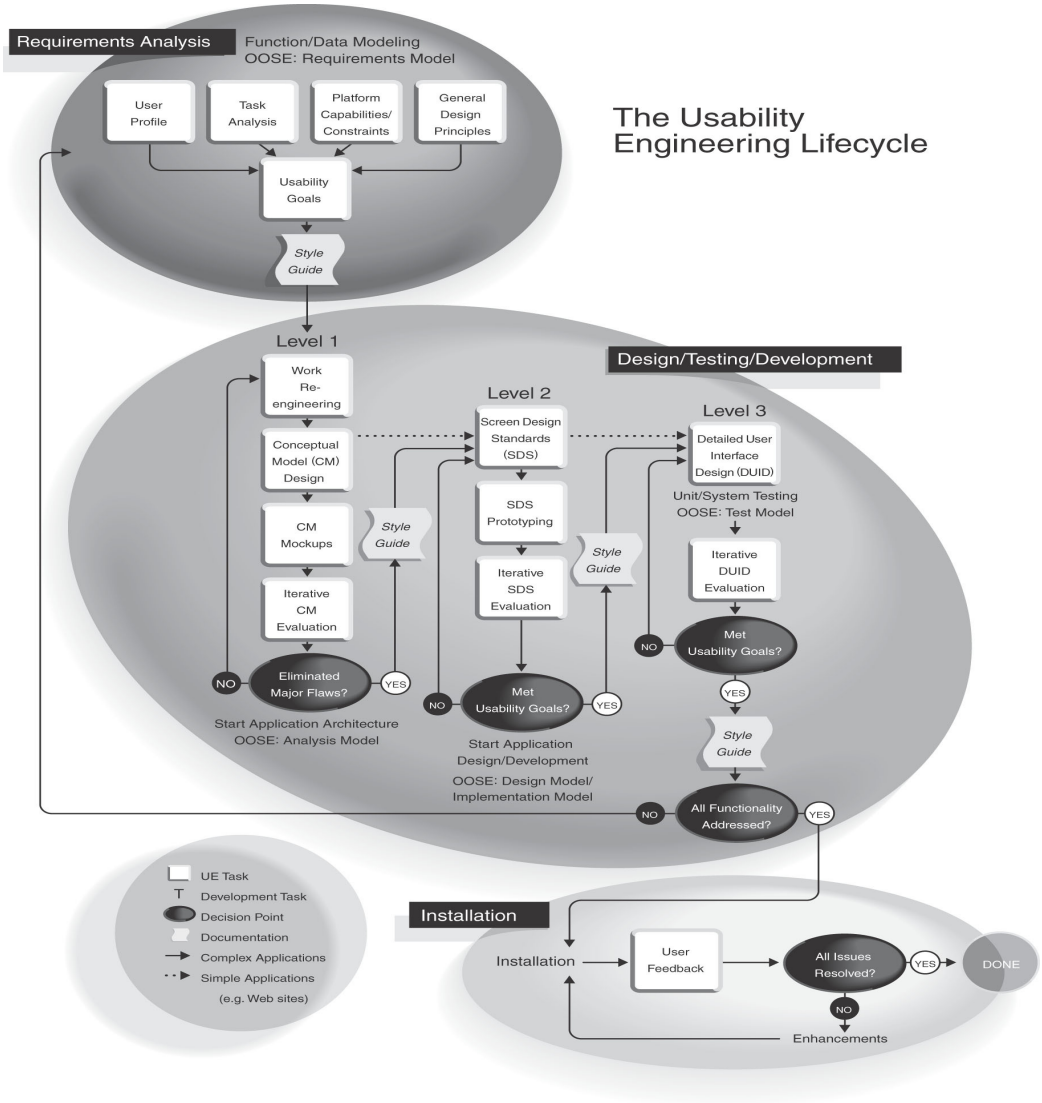

Figure 2.5: Usability Engineering Life Cycle [191]

unstructured, depending on how rigourously the interviewer abides by a prepared set of questions [163, 241]. Kitchenham and Pfleeger [163] states that in a structured interview, a fixed list of carefully worded questions forms the basis of the interview and data is usually analysed using statistical analysis. While, in a semi structured interview, the interview generally takes the form of a conversation where the interviewer starts with a set of potential topics and new questions may be posed as new information is revealed. Semi structured interviews data is usually analysed via qualitative analysis methods [163].

Kitchenham and Pfleeger [163] states that unstructured interviews are usually used to elicit scenarios while structured interviews are an efficient method for same data collection from a large number of participants. Semi structured interviews allow researchers to clarify questions and probe unexpected responses [163]. Interviews are time and cost inefficient due to the time consumed in scheduling and conducting the interviews (in person, by video conference, by phone, or over the web) [163, 241]. Interview data is usually recorded in audio or video form and thus needs to be transcribed and / or coded [163].

Focus Groups: Interviews are restricted by one person's perspective at a time. Thus *Focus groups* emerged to offer an alternative when there is a need to gather a group of stakeholders together to discuss issues and requirements [163, 241]. Focus groups can be either structured or unstructured. In the latter case, a facilitator keeps the discussion on track as and when required. In requirements elicitation, focus groups allow for acquiring consensus view and / or pinpointing conflict areas. Focus groups need to be carefully structured and administered to prevent one or few people from dominating discussions [241].

Thinking Aloud: *Thinking aloud* technique was introduced by Clayton Lewis [182] and is based on the techniques of protocol analysis proposed by Ericsson and Simon [73]. Thinking aloud is used for examining individuals' problem solving strategies that requires participants to externalize their thoughts by speaking out loudly everything that they are thinking of and attempting to do [163, 241]. The evaluator is responsible of interrupting the participant in case he falls into silence and reminding him to think out loud [163, 241]. This technique was found to be successful with children and when evaluating systems intended to be used synchronously by groups of users [241]. Thinking aloud is relatively easy to implement and manual record keeping can be used thus eliminating the need for transcription. It also provides access to participants' mental model [163].

Naturalistic Observations: Humans usually face difficulty in accurately explaining their tasks and how they achieve it. Thus all the above techniques can fall short in providing the designer with an accurate and complete picture [163, 241]. *Naturalistic observation* can complement the above techniques and provide context for tasks via involving a member of the design team to shadow a stakeholder and make notes, ask questions and observe what is performed in the natural activity context [163, 241]. The level of observer's involvement varies between no involvement (outside observation) and full involvement (participant observation) at the other end. Nevertheless, naturalistic observations requires more time and commitment from the design team [241]. In addition, it can result in enormous amounts of data that requires considerable time and effort to interpret correctly, however, observations are relatively easy to implement, require no special equipment and give fast results [163].

The above section clarifies that there exists a variety of techniques that can be used in requirements elicitation or usability evaluation in order to place users at the centre of the development process. The variety of methods provides UCD and Agile practitioners who are keen to integrate Agile and UCD with the flexibility to choose the techniques that take into consideration the project characteristics, for example, time and budget constraints and users characteristics, for example, availability, cooperation, etc.

2.6.3 Usability Evaluation Paradigms

There exists four core evaluation paradigms, quick and dirty evaluation, usability testing, field studies and predictive evaluation. Different authors and text books have slightly different terms for similar paradigms [241].

Quick and Dirty Evaluation: Preece et al. [241] declares that *quick and dirty evaluation* occurs when designers informally gather feedback from users or consultants on their ideas. This technique can be used throughout the development life cycle when fast input is required. The collected data is usually descriptive and informal and fed back into the design process [241].

Usability Testing: *Usability testing* involves measuring typical users' performance on carefully prepared system tasks while watching and recording users' performance and logging their software interactions [241]. This observational data is used to calculate performance times, compute time of task completion, identify number and type of errors and explain reasons

behind users actions [241].

Field Studies: *Field studies* are conducted in natural settings in order to understand users' tasks in their natural context and the effect of technology on them. Then this can be used to explicate the need for new technology via determining design requirements and facilitating the introduction and evaluation of technology [241].

Predictive Evaluation: *Predictive evaluations* utilise experts' knowledge of typical users to foresee usability problems. Experts are usually guided by heuristics or theoretically based models. This process is referred to as an example of discount evaluation since it is relatively inexpensive and quick since it does not require users presence or special facilities [241].

The decision for using one of the above paradigms is reliant on a number of factors: users' role, the relationship between evaluators and users during the evaluation, the evaluation location, the type of data collected and method of analysis, how the evaluation findings are fed back into the design process, and the philosophy and theory that underlies the evaluation paradigm [241].

2.6.4 Discount Usability Engineering Techniques

Discount Usability Inspection or Engineering techniques are among the most investigated and utilised techniques by both UCD and Agile practitioners to achieve Agile and UCD integration. Those techniques are repeatedly referred to in chapter 3, that reports the results of a systematic literature review that investigates the challenges, strategies and success factors for AUCDI. Those techniques are also referred to in chapter 4 that reports the results of an empirical study for industrial attempts to integrate Agile and UCD. *Discount usability inspection* is an concept coined by Jakob Nielsen to promote design and testing techniques that are low-cost and simple [211, 213] as an alternative to the high cost of traditional usability design and inspection techniques.

Simplified Thinking Aloud: *Simplified thinking aloud* is a usability testing technique used to elicit user feedback via prompting users to speak out loud their thoughts in regards to the performed tasks on the tested software and can be utilised both with paper prototypes as well as functional software [156].

Heuristic Evaluation: *Heuristic evaluation* was developed by Jakob Neilsen [211] as an informal usability inspection technique [156, 241]. It involves experts using a set of usability principles(heuristics) as guidance to evaluate user interface elements conformance to those heuristics, play the role of typical users who are using the software, and record whatever problems they encounter [241].

Rapid Iterative Testing and Evaluation (RITE): *Rapid iterative testing and evaluation method* is a discount usability testing method that differs from traditional usability testing in allowing extremely rapid changes and verification of the effectiveness of these changes. Once the usability engineer has collected data on system usability from even one participant they can make changes to the prototype according to this feedback before testing the system on the next participant. However, resources need to be dedicated in order to perform rapid changes [196].

2.6.5 Design Guidelines and Standards

Design guidelines and standards purpose is to help designers learn from others' experience and thus be able to create better designs [241]. The following section will discuss the difference between design principles, style guides and standards since they will be referred to in chapter 3, 4, 6, 7, and 8.

Design Principles: *Design principles* embody design information derived from theory and can be used to practically include cognitive models and processes into designs. An example of design principles is "recognition rather than recall", this principle is based on scientific memory related theories that claim that people find it easier to recognise things without prompting rather than remember them [241].

Style Guide: *A style guide* is a collection of specific design rules and principles that are used to ensure a consistent look and feel across a set of applications and can be used to achieve corporate images [241]. An example in apple Macintosh human interface guidelines is related to designing colour icons which states that "When you design an icon, you should start by creating the black and white version first, then the colour should be added" [241].

Standards: *Standards* are used to govern interactive systems development. The relevant standards to UCD are [241]:

1. ISO 9241 Ergonomic requirements for office work with Visual Display Terminals (VDTs).
2. ISO 13407 Human centred design processes for interactive systems.
3. ISO 14915 Design of the User Interface of multimedia.

Design principles, style guides and standards can be of significant importance in regards to integrating Agile and UCD since Agile teams vary in their team formations and in a lot of teams the role of UCD practitioner is either non existent or played by an Agile developer with experience or interest in UCD as it will be pointed out in chapter 3. This places an additional burden on Agile developers due to their lack of professional knowledge in UCD. Thus design principles, style guide and standards can provide theory and experience based design guidelines and standards that can help to achieve professional design.

2.7 User Centred Design Integration

Despite at least 20 years of research into usability engineering, a significant gap between usability work and software development remains, causing a lack of impact from usability work on software development. This lack of impact is a key challenge that faces developing high quality software [276]. Venturi and Troost [279] describes *User Centred Design Integration* as a multidimensional construct. They define it as follows:

> UCD integration is achieved when every phase of the product life cycle follows the principles of UCD, when UCD team is provided with the proper skills and experience, supported by the management commitment and a proper UCD infrastructure and awareness and culture are properly disseminated in and out of the organisation [279].

The research topic of integrating UCD into software development has been carried out for many years. This interest has led to the emergence of a set of conferences, workshops, books and journals. An example of conferences dedicated to that research topic is Human-Centred Software Engineering conference. The list of workshops held were: [25, 26, 141, 158, 159].

Seffah et al. [258] has also published a book entitled "Human Centred Software Engineering-Integrating Usability in the Development Process"; this

book discusses various aspects of the integration of usability into the development process [258]. Moreover, some journals dedicated special issues to discuss this topic, for example, "Software Process Improvement and Practice Journal "that published a special issue in April/June 2003 on "Bridging the Process and Practice Gaps between Software Engineering and Human Computer Interaction" [13].

But despite this effort and recognition, there are still difficulties in integrating those two communities, and in producing viable methods and techniques that satisfy the demands of usability practitioners and software engineers [259].

A number of reasons were identified for not integrating UCD into software development process. The integration is perceived to require new skills, awareness, commitment from the employees and if UCD is new to the organisation then this implies that training is needed [152]. Moreover, the integration requires new requirements and activities and as a result may pose an extra workload to the project that demands additional time and resources which could result in failing to abide by the project scheduled milestones [152].

Several surveys have also been conducted recently on UCD practice. For example, Rosenbaum et al. [249] survey was answered by 111 respondents from the CHI and UPA conferences and focused on a number of issues, for example, the major obstacles that prevent UCD from achieving greater strategic impact. This included resource constraints, development and management doubts about the value of UCD or usability engineering, deficiency in usability knowledge and lack of trained usability/HCI engineers [249].

In 1995 a special interest group was held entitled "Usability Management Maturity, Part1: Self Assessment-How do you Stack up?". This special interest group discussed the assessment results of UCD integration of 28 organisations over a period of five years. The findings revealed that although hiring human factors professionals, conducting usability lab tests, or involving end users in design were all key factors that contributes to the success of the integration; some organisations were successful in the integration endeavours in the absence of these key factors. This stems from blending management attention to usability issues, skilled UCD staff, and applying fundamental principles of usability to the software development process [87].

2.8 Agile and User Centred Design Integration (AUCDI)

Ungar [277] defines *Agile User Centred Design Integration* as the practice of user centred design within the wrapper of Agile software development processes [277]. This definition can be criticised for suggesting that, in order to achieve AUCDI, UCD is embedded into Agile. Different integration approaches can be followed as it will be illustrated in chapter 3.

In the past decade there has been an increased industrial and research interest in AUCDI in the Agile and UCD community. This was reflected by a number of dedicated workshops, panels, tutorials, seminars, discussion groups and publications.

Examples of dedicated AUCDI workshops are [207, 208, 252, 261, 273]. Examples of dedicated AUCDI panels are [77, 260, 261]. Examples of dedicated AUCDI tutorials are [47, 54, 230, 232, 234, 271]. Examples of dedicated AUCDI seminars are [232]. Examples of dedicated AUCDI discussion groups are Agile-usability Yahoo group and Agile UX meetup.

Examples of dedicated AUCDI publications are [3, 5, 8, 9, 20, 23, 32, 34, 36, 39, 47, 53, 55, 59, 61, 62, 71, 75, 76, 78, 80, 85, 89, 100, 113, 114, 124, 132, 153, 156, 166, 176, 183, 184, 185, 193, 197, 203, 221, 231, 251, 262, 272, 273, 277, 278].

This interest in AUCDI is arguably due to three reasons: first, the reported advantages of UCD on the developed software as it enables developers to understand the needs of the potential users of their software, and how their goals and activities can be best supported by the software thus leading to improved usability and user satisfaction. Second, the Agile community hardly discusses users or user interfaces, thus implying either a negligence of UX or focus on less sophisticated UX projects [9]. Moreover, none of the major Agile processes explicitly have guidance for how to develop usable software [178]. In addition, the interaction design role, usability, and user interface design in an Agile team is unclear and largely overlooked [20, 45]. Furthermore, principles and practices for understanding and eliciting usability and user requirements and evaluating Agile systems for usability and UX are generally considerably deficient [156, 178, 267]. This is particularly challenging for UI intensive systems since it can lead to poor usability that results in lost sales, reduced productivity, low user satisfaction and can endanger lives of human beings if it involves safety critical systems [178]. Third, there appear to be philosophical and

principled differences between Agile methods and UCD that suggest that their integration will be fundamentally challenging.

More specifically in regards to XP, Constantine and Lockwood [45] claims that XP and the other Agile processes are "light on the user side of software" and are more suitable for non GUI intensive applications. Moreover, requirements engineering is not explicitly defined within XP [221, 222]. XP has not explicated the interaction design process [82] thus devising an initial design as an activity in XP is largely not tackled [221, 222]. Dealing with usability issues in XP is dependent on on-site customer expertise [36]. Although Agile methods emphasise testing, and XP involves unit and acceptance testing, however, supportive practices for supporting usability testing are generally absent [267].

Furthermore, Jokela and Abrahamsson [153] conducted a controlled experiment to provide scientifically grounded empirical data in regards to XP and usability relationship with the purpose of comprehending the extent of guidance that XP provides for developing usable software. This experiment aimed to investigate whether XP customer centredness also results in user centredness. Thus a project that embodies a "by the book" XP implementation was examined from a usability engineering perspective. The investigation revealed that apart from some practices that imply implicit usability evaluations, XP neglects software usability related aspects. Moreover, the product's usability responsibility in XP projects is transferred to the customer. Thus if the customer is not concerned or attentive to usability related issues then the usability will be jeopardized or dependent on the usability skills and interest of the design team. The results clearly show that frequent and close cooperation between the customer and developers is not a guarantee to good usability [153].

This shortage is attributed to a number of reasons: the absence of a representative from the interaction design, usability, or human factors communities in the formation of the Agile Alliance [45]. In addition, Agile methodologies were created by industrial practitioners who focused on improving the engineering process [20] and empowering developers to meet customer demands via continuous delivery of working software [178].

It is worth noting that some Agile processes like Dynamic Systems Development Method [268, 269], Feature Driven Development [225] and Crystal [42] do specifically address design of the user interface. For instance, Feature Driven Development, has well formed user documentation and extensive on line help [102]. DSDM, has usability prototypes for exploring the user interface [268, 269], while Crystal has an explicit interaction

design process defined [102] and specifically lists UI designer and usage expert as roles in a development team [42].

In general, it is not yet clear how to incorporate UCD into Agile processes without sacrificing the acknowledged benefits of each of these individual processes. In addition, significant differences exist between Agile and UCD which create challenges to integration attempts.

2.9 Capability Maturity Models

This thesis investigates the usage of usability maturity model in the domain of integrating Agile and UCD. Thus the following sections will lay the foundation for these topics. The chapter ends by discussing usability maturity models as related to subsequent chapters.

Maturity Models are normative [140] reference models [110] that embrace the assumption of predictable evolution and change patterns. The main purpose of maturity models is to assess the current situation in order to evaluate the strengths and weaknesses and then prioritize and plan for improvement [140]. This is achieved via evolutionary successive stages or levels that signify step by step patterns of evolution and change designating the desirable or current organisational capabilities against a specific class of entities [94, 201, 248]. Those maturity levels form a path from initial state to maturity that can describe logical, anticipated, or desired evolution and change path(s) [16, 94].

A variety of maturity models with an increased quantity and breadth, in both research and practice domains, have emerged over the last years to measure and ascertain dedicated aspects of technical and social systems [201]. Examples for maturity models in different domains are: the Capability Maturity Model (CMM) [235], a widespread maturity model for software development process improvement developed by the Carnegie Mellon University software engineering institute, ISO 15504 [139], digital government maturity model [94], Business Process Management maturity model(BPM) [248], the HP business intelligence maturity model [224] and Sun (2005) Information life cycle management maturity model [270].

In the early 90's usability maturity models or usability capability assessment models were proposed to conduct a status-quo analysis of UCD. In 2000 the ISO 18529 emanated to define the UCD processes in a complying format to the ISO 15504 requirements. The UCD substance of the ISO

18529 model is based on ISO 13407.

Becker et al. [17] reported on the results of a literature review on maturity model role in research. The literature review discovered more than 1000 academic articles on maturity models in the past fifteen years. This literature review revealed that there exists an increasing interest in the topic specifically in the time period from 2005-2008 [17].

Figure 2.6 reveals the quantity of papers that discussed maturity models within the period from 1994 till 2008.

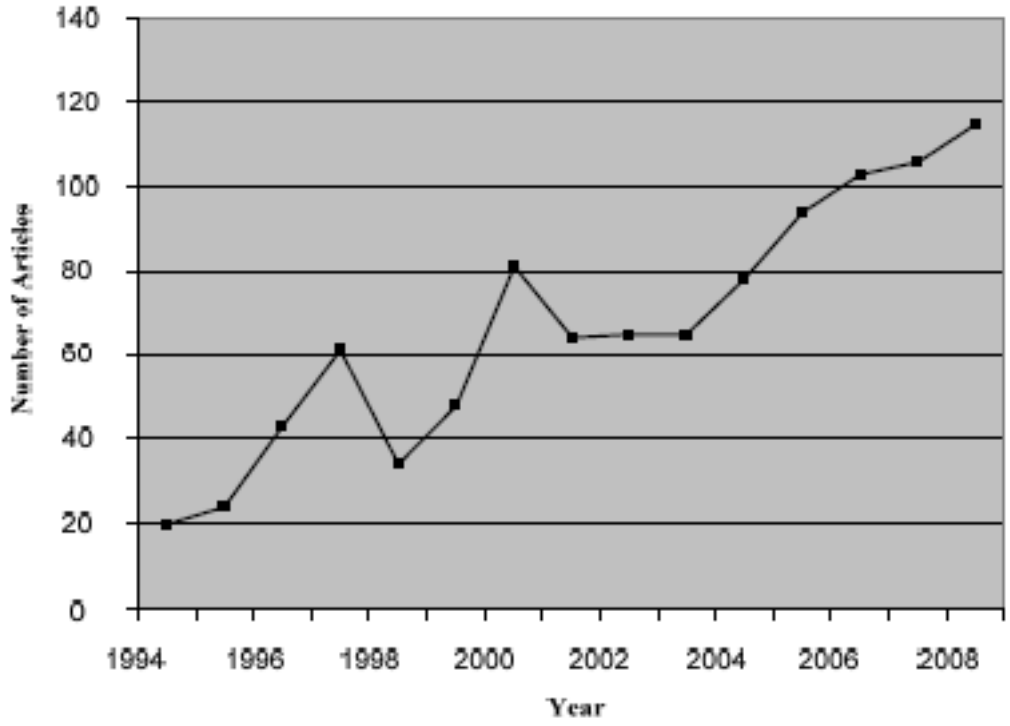

Figure 2.6: Maturity Model Papers Per Year [17]

2.9.1 Purposes of Maturity models

Maturity models have a number of purposes and these purposes should be clear to both researchers or practitioners who attempt to use a particular maturity model or evaluate it since different purposes implies different components and properties that can affect the creation, evaluation and utilisation of the models.

There exists three purposes for maturity models descriptive, prescriptive or comparative. Descriptive maturity models are used as a diagnostic tool [186] to assess the current capabilities of the examined entity against specific criteria [16]. Prescriptive maturity models aim to pinpoint desirable maturity levels and provides specific and detailed improvement guidelines [16, 186]. Finally, comparative maturity models are used for comparative purposes in external or internal benchmarking [57].

2.9.2 Critique of Maturity Models

Despite the plethora of maturity models, the concept has been subject to criticism due to poor theoretical foundation, unnecessary bureaucracy, and the falsified certainty to accomplish success [201]. King and Kraemer [160] raise concerns that maturity models should pay attention to evolution and change driving factors rather than predefined end state defined by successive maturity levels [160]. Maturity models have been criticised of being an oversimplification of reality [57]. Mettler and Rohner [202] raised concerns on maturity model negligence of multiple equally advantageous paths [202]. Iversen et al. [140] pinpointed the rigidity of some maturity models and the need for the flexibility to configure maturity models according to external and internal project and organisational characteristics [140]. Becker et al. [17] pointed out the scarcity of empirical foundation [57] and Becker et al. [17] critiqued the dis-satisfactory design process documentation.

2.10 Usability Maturity Models

A significant part of this thesis investigates the utilisation of usability maturity models in the domain of integrating Agile and UCD. Thus this section will briefly discuss the purpose of usability maturity models and then chapters 5, 6, and 7 will delve into the details of usability maturity models and its examples and will utilise two chosen usability maturity models in five case studies that integrated Agile and UCD.

Improving the effectiveness of user-centred design in software development is a considerable challenge in many organisations. *Usability Maturity Models (UMMs)* or *Usability Capability Assessment Models (UCAMs)* are methods for developing UCD processes in companies in order to facilitate

usability methodologies for creating usable products. Usability maturity models help management understand the issues surrounding organisational opportunities and improve the usability of its products. UMMs also benefit usability practitioners by pinpointing areas of improvement in usability processes and practices [151].

Usability maturity models aim to assist organisations in conducting a systematic analysis that evaluates the strengths and weaknesses of the organisation in regards to UCD related aspects [153] and accordingly plan for improvement actions [152]. Usability Capability is defined as

> A characteristic of a development organisation that determines its ability to consistently develop products with high and competitive level of usability [152].

A usability maturity model has three main elements: a UCD reference model, a performance scale and assessment guidelines [148].

1. A User centred design reference model

   Each usability capability maturity model implicitly or explicitly defines a *User Centred Design Reference Model* that presents the ideal UCD model [147]. A UCD reference model defines the elements of UCD that should be examined in an assessment. This could include the UCD infrastructure, activities, commitment and awareness, etc [147, 148].

2. A performance scale A *Performance Scale* is used to rate the elements involved in the UCD reference model, higher results indicate higher UCD maturity [148].

3. Assessment Guidelines

   Assessment Guidelines provide practical guidance for performing the assessment. These usually have notes on how to conduct the assessment, evaluate the reference model elements, interpret the findings, generate the assessment results and present it to stakeholders [148]. The assessment results can be useful in identifying weaknesses impacting UCD in order to prioritize improvement areas, identifying strengths to be protected and third party certification purposes via allowing a purchaser organisation to understand the supplier organisation maturity in developing usable products [154].

2.11 Maturity Model Development Phases

DeBruin et al. [57] proposed a set of sequential and iterative phases that can be used in developing maturity models as it can be perceived from figure 2.7. DeBruin et al. [57], Mettler [201] proposed a set of decision parameters that need to be determined in developing maturity models, those were used as the basis for developing the AUCDI maturity model as it will be illustrated in chapter 9.

Figure 2.7: Maturity Model Development Phases [57]

2.11.1 Phase 1-Scope

DeBruin et al. [57] states that the scoping phase in developing maturity models distinguishes the proposed model from other models and determines the model's specificity and extensibility. In addition, the combination of scoping decisions will set the boundaries and limitations for the model's use and application [57]. Both DeBruin et al. [57], Mettler [201] stated that there are a number of decision parameters that need to be determined in scoping maturity models. Those included focus, level of Analysis/depth, development stakeholders [57, 201] and dissemination [201].

Focus: The determination of the maturity model scope helps in setting outer boundaries for its application and use, distinguishing the proposed model from other existing models and determining the specificity and extensibility of the model.

Level of Analysis: The level of analysis helps in determining the model's depth and conditions the model's operating altitude [201].

Development Stakeholders: Determining the development stakeholders is of significant importance since they can contribute to the model development process [201].

Dissemination: The dissemination determines whether the model will be open to specified audience or will be available for exclusive access only.

2.11.2 Phase 2 - Design

The second phase in developing maturity models involves determining the model design where a number of decisions related to the intended model audience needs to be made. These decisions are: the purpose that the model audience seek to achieve via applying the model, the method of applying the model, individuals who need to be involved in applying the model and what can be achieved through applying it [57]. Moreover, Mettler [201] detailed the pivotal work of DeBruin et al. [57] by presenting particularized decision parameters for designing maturity assessment models. Mettler [201] added maturity definition, goal function, design process and design product as additional decision parameters to those suggested by DeBruin et al. [57].

The decisions parameters related to maturity model design that are proposed by DeBruin et al. [57], Mettler [201] are as follows

Method of Application: The method of application is related to whether the data is collected based upon a third party or self assessment [57].

Driver of Application: The driver or application can be either internal or external or both [57].

Respondents: The respondents are the parties that are contacted for data collection [57].

Application: The application is related to the entity, region, organisation unit etc. to which the model is applied [57].

Maturity Definition: Mettler [201] indicates that maturity definition involves defining what entails 'maturity'. The maturity definition can be process focused, object focused, people focused or a combination. A process focused maturity concentrates on defining more effective procedures via focus on work practices and activities (e.g., specified tasks inputs and outputs). While, an object focused maturity aims to enhance application mode via investigating work products' features (e.g., non functional and functional requirements). Finally people focused maturity is concerned with people's soft capabilities (e.g,'s skills, proficiency, people's feelings and behaviour) [201].

Goal Function: The goal function of maturity models is concerned with how maturity is progressed and represented. Maturity representation involves deciding whether the maturity progress is one dimensional, where the entire focus is on a single target measure (i.e., efficiency) or multidi-

mensional, where the entire focus is on multiple, sometimes conflicting goals or competitive bases [201].

One dimensional linear stages results in an average maturity stage for the assessed entity. Thus it cannot adequately represent the maturity of complex domains. Moreover, it does not provide adequate or sufficient guidance to organisations that aim to improve the as-is situation. While multidimensional maturity models is capable of providing organisations with a more profound understanding to their strengths and weaknesses and pinpointing areas that need improvement and consequently allowing for more efficient resource allocation. Finally, multidimensional models allows drilling down and tailoring the assessment report to the varying needs of multiple audiences [57].

Design Process: The nature of the design process (e.g., practitioner based versus theory driven or a combination of both) should be determined so as to identify the knowledge base utilised for deriving the maturity levels, the maturity metrics and the improvement recommendations. Moreover, this decision heavily influences the research methods to be used (e.g., focus group discussions versus theory driven) and it also affects the practical and scientific quality of the resultant design product [201].

Design Product: The quality of the practical usage of the resulting design product (maturity model) is also affected by the design product shape (pure textual description, functioning of the maturity model, instantiation as a software assessment tool) [201].

Maturity Stages: The design phase also involves decisions in regards to maturity stages. DeBruin et al. [57] declares that the definition of maturity stages can be achieved via either a bottom-up or top-down approach. Bottom-up approach involves starting by pinpointing requirements and measures and then writing definitions to reflect these [57]. Whereas, top-down approaches involves starting by writing stage definition and then measures are developed to fit the stage definitions. This approach works well with relatively naive domains where there is scarce evidence of what represents maturity. Thus the emphasis is on what embodies maturity and then how to measure it.

2.11.3 Phase 3 - Populate

The populate phase focuses on determining the model content via focusing on what needs to be measured and how to measure it via identifying domain components and sub components. A domain component is a major, independent aspect that is significant to a particular domain maturity e.g. critical success factors, barriers to entry. These domain components are utilised in general stage definitions and in results clustering for model audience. Pinpointing the domain components allows the identification of specific improvement strategies. In a mature domain an extensive literature review can be used to identify the domain components. While, in a relatively new domain existing literature may not be enough to derive a comprehensive domain components list. As a result, a literature review is perceived as sufficient as a theoretical starting point and another way of identification is needed, for example, interviews to validate the a priori-constructs and increase the established collectively exhaustive and mutually exclusive list of critical success factors. The confirmation of components chosen from several sources of evidence leads to improving the extensibility of the final maturity model findings [57].

DeBruin et al. [57] states that domain sub components assist in the development of assessment questions used in the maturity questionnaire, enable richer analysis of maturity results, represent specific capability areas that enable targeted maturity level improvements, and improve the ability to present maturity results in order to meet the needs of target audience. The goal is to reach domain components and sub-components that are collectively exhaustive and mutually exclusive [57].

2.11.4 Phase 4 - Test

After the population of the maturity model it should be tested for rigour and relevance. Testing focus should fall on two aspects: the model's construct and the model instruments for reliability, validity and generalizability [57].

2.11.5 Phase 5 - Deploy

DeBruin et al. [57] states that the model generalizability will remain questionable until the model is deployed to entities independent of the devel-

opment and testing activities. For specific domain models where single organisational stakeholders were involved, it is necessary to identify similar firms in different markets in order to prepare a list of potential "next" administrations. For general domain models where multiple organisational stakeholders were involved in model development, it is recommended to use a consortium for further model application. The final steps to achieve global standardization and acceptance of the model involves the identification of organisations that may benefit from future utilisation of the maturity model and the ability to apply the model to multiple entities.

2.11.6 Phase 6 - Maintain

Success in establishing the model generalizability depends on assessing the model's ability to deal with a high volume of model applications. This requires the presence of a repository to track model development and evolution as the understanding of the domain knowledge and the model deepens and broadens.

2.12 Chapter Summary

This chapter provided an overview of foundational concepts related to UCD and Agile development processes. Understanding Agile values and principles together with UCD principles and activities are key factors to achieve AUCDI. Moreover, there exists a variety of proposed activities and principles for UCD that reflect that UCD is more of a philosophy that places users at the centre of the development process rather than dictating a rigid approach on how to achieve that. Finally, the chapter discussed maturity models and a key point to get out of the chapter is that there is an absence of an AUCDI maturity model or assessment model, a research gap that chapter 8 will attempt to fill by proposing a maturity model for integrating Agile and UCD that was developed via taking into consideration the purposes, critique and design principles of maturity models that were discussed in this chapter.

The next chapter will detail a SLR that focused on Agile and UCD integration.

Chapter 3

Systematic Literature Review

This chapter provides details of a Systematic Literature Review (SLR) that was conducted on Agile and UCD integration. The aim of this SLR was to identify various challenging factors that restrict AUCDI and explore the proposed practices or success factors to deal with them. The chapter commences by discussing the research method used for conducting the SLR for AUCDI. It discusses the quantitative and qualitative results, then it justifies the need for integrating Agile and UCD via highlighting integration advantages. This is followed by an overview of the differences and commonalities between Agile and UCD that could respectively act as impediments and facilitators for the integration. The subsequent sections present the results of the research questions in regards to AUCDI challenges, practices and success factors and is followed by a conclusion section.

3.1 Research Methodology

Kitchenham and Charters [161] defines systematic literature reviews as

> A systematic literature review is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest. Individual studies contributing to a systematic review are called primary studies; a systematic review is a form of secondary study.

A SLR utilises explicit, rigourous and systematic techniques in order to identify, critically appraise, and synthesize primary studies focused on a certain subject. This is achieved by developing and documenting a SLR protocol that can be used by other researchers who attempt to critically appraise and replicate the results. SLR strength is signified by its explicit and rigourous attempt to minimize mistaken or misleading conclusions that could arise from biases in either the primary studies or the review process itself [64].

Features of Systematic Literature Reviews

The following features differentiate a systematic literature review from conventional literature review [161]:

- SLRs define a review protocol that explicates the addressed research question and the methods that will be utilised to execute the review.
- SLRs are founded on a defined search strategy that aims to identify as much as possible of the relevant literature.
- SLRs document their search strategy so as other researchers can assess the SLR rigour and completeness and the process repeatability.
- SLRs require explicit inclusion and exclusion criteria to assess each potential primary study.
- SLRs specify the desired information to be obtained from each primary study including quality criteria that is used to evaluate each primary study.
- SLR is a prerequisite for quantitative meta-analysis.

Evidence based software engineering (EBSE) enables software practitioners and academics to make informed decisions via integrating methodological research rigour with practice [66, 162]. SLRs are a pivotal tool for

evidence based practice since they merge multiple studies' findings and focus on the production of generalizations via empirical research integration [22].

3.1.1 Systematic Literature Review Protocol Development

This thesis aims to use a rigourous and auditable methodology in conducting the literature review in order to minimize researcher bias. Thus the SLR was based on the guidelines presented in [22, 161].

Our objectives from conducting this systematic literature review were:

- To identify existing evidence regarding challenges involved in the AUCDI integration.
- To infer relevance of AUCDI challenges to situational characteristics related to software development settings.
- To identify AUCDI success factors and strategies.
- To infer relevance of AUCDI success factors and strategies to AUCDI challenges and situational characteristics related to software development settings.

Prior to performing the SLR, the planned SLR protocol was evaluated via two methods as recommended in [22]. The first method involved asking experts to review the protocol; in this case PhD supervisors served this role. The second method involved testing the protocol execution on a reduced set of selected sources. If the obtained results are not suitable, the protocol was reviewed and a new version was created. This piloting of the SLR is reported in section 3.1.5.

3.1.1.1 Background

AUCDI literature contains only two literature reviews that have been reported so far. The first is a literature review that discussed methods for integrating usability engineering practices into the Agile software development process and identified the tensions between Agile methods and usability engineering [267]. The second is a SLR that revealed the existence of a common process model for integration and discussed the supporting artifacts for the collaboration between designers and developers [53].

Thus there is an absence of a SLR that provides a comprehensive scrutiny of AUCDI challenges and investigates the success factors and practices that tackle these challenges. The results of this analysis can be used by organisations that aim to achieve the integration in order to understand the potential challenges involved in the integration and the available practices that can be utilised to tackle these challenges.

3.1.1.2 Specifying the research question(s)

Research questions are the main driver to the SLR methodology. A substantial domain analysis of literature was conducted that focused on previous AUCDI attempts. This resulted in identifying the following research questions that were raised by researchers addressing AUCDI.

- When and how to conduct user research in Agile projects in a manner that fits the tight Agile time lines?
- How to design user interaction in Agile projects?
- How to evaluate usability and UX of software developed in Agile projects?

Our study has a different scope and attempted to address the following questions whose answer can play a significant role in formalizing and structuring AUCDI efforts.

- What are the challenges that could develop during AUCDI adoption process?
- What are the potential success factors for AUCDI?
- What are the potential practices for AUCDI?

3.1.2 Search Process

This section details the process that was followed to search for literature. This search process aimed to identify as much as possible of the relevant literature and document the search strategy so as other researchers can assess the SLR rigour and completeness and the process repeatability. The search process identifies: the search resources, search keywords, inclusion and exclusion criteria, papers classification and citation management and retrieval.

3.1.2.1 Search Resources

The search included electronic sources, conference proceedings, journal articles and magazines. The search string focused on combining both UCD and Agile keywords and was modified in accordance to the specific search requirements of the different electronic libraries.

The electronic sources/digital libraries chosen to conduct the search on were:

- ACM Digital Library, http://dl.acm.org/
- IEEExplore Digital Library, http://ieeexplore.ieee.org/Xplore/
- Google Scholar, http://scholar.google.com
- Springer, http://www.springerlink.com/
- Wiley InterScience, www.interscience.wiley.com
- Citeseer Library, http://citeseer.ist.psu.edu

Conference Proceedings

In addition, the following conferences proceedings were manually searched for research papers and experience reports on the topic. The search covered the time period from 2000 till 2012. However, some conferences did not exist until after the year 2000 and other conferences were no longer in existence before the year 2012. Full details regarding the start and end search date of every conference are shown in table 3.5.

- Agile Conference
- XP
- XP/Agile Universe
- International Conference on Software Engineering (ICSE)
- Human Factors in Computing Systems Conference (CHI)
- International Symposium on Empirical Software Engineering and Measurement(ESEM)
- British HCI
- NordicHCI
- INTERACT

- European Conference on Cognitive Ergonomics

Journals

The following journals were manually searched for papers:

- Empirical Software Engineering
- Software Practice and Experience
- International Journal of Human-Computer Studies
- International Journal of Human-Computer Interaction
- Behavior and Information Technology
- Information and Software Technology
- IEEE Transactions on Software Engineering
- ACM Transactions on CHI
- Human Computer Interaction Journal
- Interacting with Computers

Magazines

- IEEE Software
- Communications of the ACM
- Interactions

The references of primary studies were also checked for any relevant studies irrespective of the forum of publication.

3.1.2.2 Search Keywords

The research questions in section 3.1.1.2 were used to identify the search keywords that is to be used for conducting the SLR. Table 3.1 lists the keywords used for conducting the SLR

The search string used was as follows:

" "Usability" OR "User Experience" OR "User Centred Design" OR "User Interface" OR "User Interaction" OR "Usability Engineering" or "Human-Centred" AND " "Agile" OR "Agile Method" OR "Agile development" OR

Category	Keywords
UCD	Usability User Experience User Centred Design User Interface User Interaction Usability Engineering Human-Centred
Agile	Agile Method Agile Development Agile Practice Agile Project Scrum Extreme Programming

Table 3.1: Keywords for Systematic Literature Review Process

"Agile Practice" OR "Agile Project" OR "Scrum" OR "Extreme Programming"

3.1.2.3 Study Selection Criteria (Inclusion and Exclusion Criteria)

The following section discusses the criteria that were used to assess each paper and decide on whether to include or exclude primary studies.

Inclusion Criteria

To decide on paper inclusion, the following features must exist on the paper

- Peer reviewed to ensure quality of primary study.
- Available on-line to ensure paper accessibility.
- Focused on the integration of UCD and Agile to ensure its relevance.
- Focused on Scrum or Extreme Programming or Agile processes in general.
- Not a workshop, panel, tutorial, seminar, interview or poster session to ensure enough details are included in the paper in order to answer research questions and assess the paper's quality.
- Published from the period between the year 2000 and 2012.

- Written in English.
- Non redundant since the main focus was to study AUCDI challenges, practices and success factors. Thus a decision was made to exclude all papers written by the same authors that discuss the same AUCDI practices and success factors.

3.1.2.4 Papers Classification

SLRs specify the desired information to be acquired from each primary study. Thus papers were classified following the recommendations in [22] according to both descriptive and content information.

Descriptive Classification

Papers were descriptively classified according to the Agile method used and the study type used. In regards to the Agile method used, four different categories were used: XP, Scrum, Agile and mix of XP and Scrum. The inclusion of multiple studies in SLRs helps in identifying the consistency and generalization of findings across settings [64] thus other study types besides empirical studies were included, for example, theoretical studies, literature reviews and systematic literature reviews. Thus the study type classifications were as follows:

1. **Theoretical Papers:** are papers that discuss reflections in regards to AUCDI strategies, challenges or success factors without including any validation on these ideas.
2. **Empirical Papers:** are those papers that investigate, describe, predict, and explicate a phenomena via utilising observation or experience based evidence [264]. The type of empirical research were as follows:
    - **Controlled Experiment:** are papers that attempt to examine causal processes and relationships in order to explain the occurrence of a particular phenomenon [286].
    - **Surveys:** are papers that utilise surveys to deal with situations when it is not desirable or possible to control the independent and dependent variables. Surveys are used when there is a need to study the natural settings of a particular phenomena that occurred in the recent past or current time [264].

- **Case Studies:** are papers that examine a contemporary phenomenon in its real life context specifically when the context and phenomenon boundaries are not apparent [286]. This type of study was divided into single or multiple case studies according to the number of case studies reported per paper. The case studies could utilise a number of research methods including: observation, interviews, document analysis, and artefact analysis.
- **Tools:** are papers that discuss proposed tools that assist the process of integrating Agile development processes and UCD.

3. **Literature Reviews:** are papers with the sole purpose of reporting the results of reviewing the AUCDI literature.
4. **Systematic Literature Reviews:** are papers with the sole purpose of reporting the results of a systematic review of the AUCDI literature.

Content Classification

Content of papers was classified according to the integration approach used and result.

1. **Integration Approach**

   The integration approach was used to classify AUCDI papers into a number of categories that investigate one of the following topics:

   - Integrating Agile and UCD as two separate processes.
   - Incorporating UCD techniques into Agile development process.
   - Adapting or extending Agile practices to take UCD into account.
   - Adapting or extending organisational practices to suit AUCDI
   - Adapting or extending UCD techniques to suit Agile development process.
   - Proposing a tool support for the integration.
   - Introducing new team roles.
   - Investigating developers and UCD practitioners engagement.
   - Reporting the impact of AUCDI on team.
   - Reporting the impact of AUCDI on product.

   Some papers had multiple classifications.

2. **Results**

   The results of each paper were classified into two categories: success or failure in integrating Agile and UCD.

Biolchini et al. [22], Kitchenham and Charters [161] recommends classifying SLR papers via reading papers' title and abstract, however, Brereton et al. [31] points out the poor quality of abstracts thus he advises to review conclusions as well in order to decide upon primary paper inclusion. As a result, two steps were used in classifying the papers: the first step involved searching for the primary papers via utilising the search process and then reading the title, abstract, and conclusion of each paper. The second step involved examining the satisfaction of the inclusion/exclusion criteria.

All studies were evaluated by myself. In case of any classification uncertainty, a meeting was held involving myself and PhD supervisors until consensus was reached.

Exclusion Criteria

Any papers that does not possess any of the inclusion criteria was excluded. The remaining papers were read fully. A list of included and excluded papers was kept. Results bias was attempted to be avoided by excluding multiple publications of similar research. This led in some cases to contacting authors to verify the most complete and recent publication. Workshops, panels, tutorials, seminars, interviews and poster sessions were also excluded since they are usually short papers that do not provide enough details to either assess the paper's quality or answer the research question.

Since this SLR is part of a PhD thesis it was reviewed by PhD supervisors. This evaluation involved checking the internal protocol consistency to confirm that:

- The research questions discussed in section 3.1.1.2 derive the search strings discussed in section 3.1.2.2.
- The data extraction forms allow answering the research question(s).
- The data analysis procedure is adequate to address the research questions.

Citation Management and Retrieval

Our citation management procedure involved entering relevant citations

into Jabref, an open source bibliography reference manager. Then Microsoft Excel was used to record the following:

- The citation source
- The inclusion/exclusion decision
- The quality evaluation decision

3.1.3 Study Quality Assessment

Evidence based research considers SLRs pivotal for selecting and combining findings from relevant primary studies [264]. Since the search can involve digital libraries, electronic sources, conferences proceedings, etc., this can lead to enormous amount of material with extreme variance of research quality. Since the quality of evidence provided by SLRs is dependent on the quality of primary studies thus the inclusion and exclusion criteria should be accompanied by a rigourous assessment for the quality of primary studies [64]. This can help the researcher to perform a more thorough inclusion/exclusion criteria, scrutinize whether quality differences explain results differences, weigh the individual studies' importance when synthesizing results and guide further research recommendations [161].

There is an absence of an agreed upon definition of "quality" and appraising the published research quality is problematic since conference papers rarely provide sufficient detail of the utilised research methods due to space limitations of conference proceedings. Thus what is subjected to quality assessment could be the reporting quality rather than the research quality [64]. Moreover, the quality of the primary studies in the domain of software engineering is often poor [164, 264].

The quality of a study is usually assessed via a number of aspects including the rigour, credibility, reporting and relevance of the study. This involves checking the design, conduct and analysis of the primary study to ensure that the researchers have attempted to prevent systematic bias or errors [64, 161]. The result of the study quality assessment is used to decide whether to include or exclude the paper as it will be illustrated in section 3.2.2.1 and in tables B.1 and C.1 that discusses quality assessment for research papers and experience reports respectively.

Evaluating the Quality of Primary Studies

The data set of primary studies was assessed according to a number of quality criteria. These criteria were informed by those proposed for the Critical Appraisal Programme (CASP) (http://www.phru.nhs.uk/Pages/PHD/CASP.htm), in particular those for assessing the quality of qualitative research and by [63, 64, 96, 117, 164] which discussed quality assessment of empirical studies and principles of good practice for conducting empirical research in software engineering. After piloting the SLR the quality criteria were changed and divided into two different lists of quality criteria: one for experience reports and the other for academic research. This was due to the lack of rigour of reporting industrial experience reports. However, since the focus was to answer the research questions regarding AUCDI challenges, success factors and practices and benefit from reported literature that reflects both views from academia and industry thus it was decided to have different quality criteria for judging the industrial experience reports and the academic research papers.

Although these sources used a dichotomous ("Yes" or "No") grading scale, however, the desire to have a finer grained view of any attempt for achieving quality resulted in using a three point scale ("No attempt", "Attempt", "Full Attempt").

The criteria covered four focal quality related aspects that are pivotal in evaluating the primary studies [64, 65].

- Reporting: These criteria focus on the quality of reporting the motivation, aims and study context.
- Rigour: These criteria focus on the rigour of research methods that can impact the trustworthiness of the findings. It is related to the data collection and analysis validity.
- Credibility: These criteria are related to assessing that research findings are meaningful and well presented.
- Relevance: These criteria are related to judging the study relevance to the research community in specific and software industry in general.

Section 3.1.3.1 discusses the quality criteria that were used to determine the inclusion and exclusion of identified research papers. The results of applying these quality criteria on papers is illustrated in Appendix B.

3.1.3.1 Research Papers Quality Criteria

The following quality criteria was used to judge the quality of research papers this list was developed iteratively with PhD supervisor through the piloting phase.

1. Are research aim(s)/ research question(s) clearly defined?
2. Is study context adequately described?
3. Is the study based and linked to literature or practice?
4. Is research design appropriate to address the aim(s)/ research question (s) of research?
5. Is research method sufficiently described?
6. Was data collected in a way that addresses the research issue (data sources, collection, storage, validation)?
7. Are data analysis methods adequately described?
8. Are threats to validity addressed in a systematic way?
9. Are ethical issues addressed properly (personal intentions, integrity issues, consent, review board approval, relationship between researcher and participants)?
10. Is there a clear statement of findings with credible results and justified conclusions?
11. Is future research reported suitably for its audience?

Section 3.1.3.2 discusses the quality criteria that were used to determine the inclusion and exclusion of identified experience reports. The results of applying these quality criteria on papers is illustrated in Appendix C.

3.1.3.2 Experience Reports Quality Criteria

1. Are research aim(s)/ research question(s) clearly defined?
2. Is context adequately described?
3. Is the study based and linked to literature or practice?
4. Is proposed approach sufficiently described including requirements, pros and cons?

5. Is there a clear statement of findings with credible results and justified conclusions?

These quality criteria were used as the basis for the inclusion/ exclusion of the primary studies based on the evaluation of its quality.

3.1.4 Data Extraction Strategy

Data extraction strategy focuses on defining the procedures used to obtain information required from primary studies. These information are related to answering the raised research questions and identifying the descriptive and content related information. In those cases where the data have been manipulated or any inferences or assumptions were made about the papers then this should be both specified and appropriately validated by the review protocol. Data extraction involves the design of data extraction forms to ensure that sufficient and appropriate data is collected to address both the research questions and the quality criteria [161]. Biolchini et al. [22] stated that there are two kinds of results that could be extracted from the selected primary studies. These results can either be objective, in which case results are directly extracted from primary studies or subjective, in which case results are not extracted directly from the selected primary data sources but rather are extracted through communicating with papers' authors or via forming general impressions and abstractions by the SLR author.

During this stage, predefined extraction forms were used to record data extracted from reading primary studies.

Data extraction consistency was checked via two methods: first, the data extraction was carried out by myself via randomly selecting a sample of primary studies and subjecting them to data extraction by PhD supervisors. The results were cross checked and any disagreements were discussed and resolved in meetings. Second, a test-retest process was conducted by myself where primary studies were randomly selected and a second extraction was performed to check data extraction consistency.

3.1.5 Piloting the Systematic Literature Review Protocol

Piloting the SLR protocol is essential to identify errors in the procedures of data collection and aggregation [31]. When the SLR protocol was piloted the search strings had to be adapted according to the different requirements of digital libraries and electronic sources. The pilot study led to removing two electronic libraries from the original list of search resources since the University of York did not have a subscription there.

- SCOPUS (http://www.scopus.com/)
- Compendex EI (www.engineeringvillage2.org/)

Searching the digital libraries and electronic sources resulted in an enormous number of records that were extremely difficult to handle. As a result, a one to one session was conducted with the library liaison of computer science department at the University of York to provide tips on conducting more efficient search. This SLR started with 14 criteria for evaluating the quality of the empirical studies informed by those proposed in [63, 64, 96, 117, 164]. The lack of rigour of reporting industrial experience reports made it impossible to assess some of those 14 criteria. Since the SLR focus was to answer the research questions regarding AUCDI challenges, success factors and practices and benefit from reported literature that reflects both views from academia and industry thus it was decided to have different quality criteria for judging the industrial experience reports and the academic research papers. The same approach was utilised in da Silva [52]. However, the quality criteria of this SLR differed from that in da Silva [52].

3.1.6 Data Synthesis Method

Data synthesis is reported to be the single most challenging task of performing a systematic literature review since it involves scrutinizing primary papers that involve a diversity of research methods and theoretical perspectives [50]. The data was investigated via thematic analysis; a data synthesis method that identifies, analyses and reports patterns (themes) within data. It describes and organises the data set in rich detail and interprets different aspects related to the research topic [30]. The data was extracted via following the iterative thematic synthesis process recommended in Braun and Clarke [30] as shown in figure 3.1.

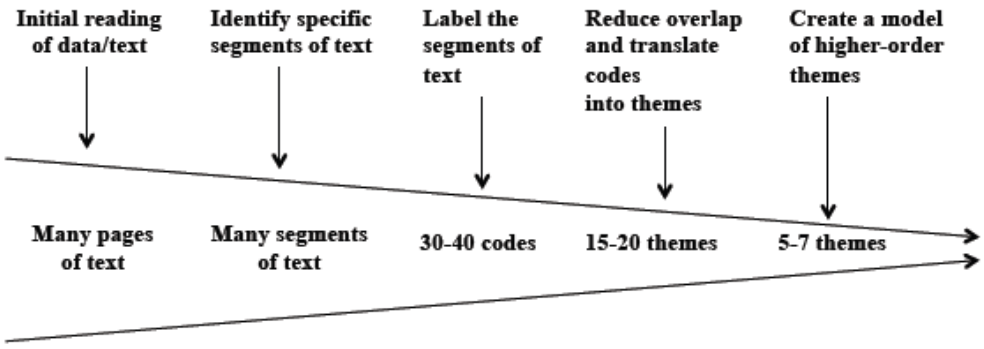

Figure 3.1: Thematic Synthesis Process [50]

The steps recommended in [50] were followed to perform the thematic synthesis as follows:

1. **Data Extraction**

   Data extraction commenced by reading all primary papers to get immersed in data, then specific text segments were identified pertaining to the objectives of synthesis, this was followed by extracting bibliographical, descriptive and content information as well as information to answer research questions. Then the extraction was checked by PhD supervisors to ensure consistency. Finally a test retest process was performed by myself to check data extraction consistency.

2. **Data Coding**

   Relevant concepts, categories, findings and results were systematically identified and coded across the entire primary studies data set. This was performed via labeling and coding important text segments like concepts, categories, findings and results. This was followed by coding across the entire data set done on a level that is appropriate for the research questions. The integrated approach in data coding was used which employs both deductive(start list) as well as inductive (ground up) coding [50].

3. **Translating Code into Themes**

   Cruzes and Dyba [50] declared that the definition and analytic function of a theme varies among researchers, and the term theme is often used interchangeably with other terms, for example, domain, category, unit of analysis, phrase, etc. Cruzes and Dyba [50] mentioned that themes condense large code amounts into a smaller amount of

analytic units, and assist the researcher to evolve more integrated schema for understanding interactions and local incidents. In this step codes were analysed and this resulted in relabeling codes, dropping redundant codes and merging similar ones. Then codes were translated into themes.

4. **Creating a Model of Higher Order Themes**

   The relationships between themes were explored and a model of higher order themes was created by checking that themes have been compared across studies, translated into each other and interpreted into higher order themes. Then higher order themes and the relationship between the themes were checked against the research questions of the synthesis. This was followed by scrutinising the presence of clear descriptions of the higher order themes and the relationships between them.

3.1.7 Documenting the Search

SLRs should be transparent and replicable and that is the driver behind the thorough documentation, recording and justification of any changes that we followed. Table 3.2 lists the procedures followed for documenting the search process. The recommendations of search process documentation that were followed was given in Kitchenham and Charters [161].

Data Source	Documentation
Digital Libraries	Name Date of Search Years Covered by Search
Journals and Magazines	Name Start Date and Volume Number for Years Searched End Date and Volume Number for Years Searched Unsearched Issues
Conference Proceedings	Title Start Search Year End Search Years Unsearched Proceedings
Efforts to Identify Unpublished Studies	Research Groups Contacted Researchers Contacted Research Web Sites Searched (Date and URL)
Efforts to Identify Unclear Information	Research Groups Contacted Researchers Contacted Research Web Sites Searched (Date and URL)

Table 3.2: Search Process Documentation [161].

3.2 Results

This section will discuss both the quantitative and qualitative results of the SLR. This section starts with an overview of search sources that discusses details of searched digital libraries, journals, magazines and conference proceedings. Then it moves into discussing the efforts exerted to identify missing or unclear information. Then it explicates the studies' overview and discusses the excluded papers and the reasons behind the exclusion. Then it discusses the results of studies classification.

3.2.1 Search Sources Overview

A number of digital libraries were searched: ACM Digital Library, IEEExplore Digital Library, Google Scholar, Springer, Wiley InterScience, and Citeseer Library. The date of the search was between the period of April to September 2012 and covered the years between 2000-2012 since papers on Agile development processes started around the year 2000 but papers regarding the integration between Agile and UCD started shortly after that. Since the aim of the SLR was to cover as many papers as possible, papers

from 2012 were included as well, however, not all papers published in 2012 were included since the search took place in the time interval between April 2012 till September 2012.

Table 3.3 lists the details of the manually searched journals. This list included the start date and start volume number and the end date and the end volume number searched. Table 3.4 lists the details of the manually

Name of Journal	**Start Date and volume Number for Years Searched**	**End Date and Volume Number for Years Searched**
Empirical Software Engineering	Volume 5, Number 1, March 2000	Volume 17, Issue 4-5, August 2012
Software Practice and Experience	Volume 30, Issue 1, January 2000	Volume 42, Issue 7, July 2012
International Journal of Human-Computer Studies	Volume 52, Issue 1, January 2000	Volume 70, Issue 9, September 2012
International Journal of Human-Computer Interaction	Volume 12, Issue 1, January 2000	Volume 28, Issue 7, July 2012
Behavior and Information Technology	Volume 19, Issue 1, January 2000	Volume 31, Issue 6, June 2012
Information and Software Technology	Volume 42, Issue 1, January 2000	Volume 54, Issue 9, September 2012
IEEE Transactions on Software Engineering	Volume 26 , Issue 1, January 2000	Volume 38, Issue 3, March 2012
ACM Transactions on CHI	Volume 7 Issue 1, March 2000	Volume 19 Issue 1, March 2012
Human Computer Interaction Journal	Volume 15, Issue 1, 2000	Volume 27, Issue 1-2, April 2012
Interacting with Computers	Volume 12, Issue 3, January 2000	Volume 24, Issue 2, March 2012

Table 3.3: Manually Searched Journals

searched magazines. This list included the start date and volume number and the end date and volume number searched.

Table 3.5 lists the details of the conference proceedings that were manually searched. This list included the start and end years searched. As it can be seen from table 3.5 that ten conference proceedings were manually searched. The earliest year for included conference proceeding is 2000 and the latest is 2012. Although our target was to search and include all conferences from the year 2000 till the year 2012, however, some conferences

Name of Magazine	Start Date and volume number for Years Searched	End Date and Volume number for Years Searched
Interactions	Volume 7, Issue 1, January and February 2000	Volume 19, Issue 3, May and June 2012
IEEE Software	Volume 17 , Issue 1, January and February 2000	Volume 29 , Issue 3, May and June 2012

Table 3.4: Manually Searched Magazines

Name of Proceedings	Start Search Year	End Search Year
Agile Conference	2003	2011
XP	2003	2012
XP-Agile Universe	2002	2004
International Conference on Software Engineering (ICSE)	2000	2011
Human Factors in Computing Systems Conference (CHI)	2000	2011
International Symposium on Empirical Software Engineering and Measurement(ESEM)	2007	2011
British HCI	2000	2010
NordicHCI	2002	2010
INTERACT	2001	2011
European Conference on Cognitive Ergonomics	2006	2011

Table 3.5: Conference Proceedings Searched

have not started at the year 2000. For example, Agile conference and XP conference started in 2003, NordiCHI started at 2000 but the proceedings available on-line started from the year 2002, ESEM and INTEACT started in 2007 and 2001 respectively. In regards to the European conference on cognitive ergonomics although it started in 2000 yet it was named "The annual conference on European association of cognitive ergonomics" and it was not possible to find the electronic proceedings for the period from 2000 till 2005. In addition, at the time of conducting the search, some of the proceedings for the year 2012 were not available on-line yet. All this led to the inclusion of some of the conference proceedings until the year 2011 rather than 2012. Some conferences were biennial, for example, NordicHCI in even years and INTERACT in odd years.

Table 3.6 lists a sample of the details of efforts exerted to complete missing information and studies via contacting the authors of the papers or other researchers. Due to space limitations the following sample was only included. Full details on efforts exerted to identify missing studies were recorded.

Table 3.7 shows a sample of the details of efforts exerted to identify unclear information via contacting the authors of different papers. Due to space limitations the following sample was only included. Full details on efforts exerted to identify unclear information were recorded.

3.2.2 Studies Overview

Section 3.1.2.3 lists the inclusion and exclusion criteria that were utilised to decide on inclusion or exclusion of papers. A total of 80 papers were excluded for various reasons as it will be illustrated in section 3.2.2.1.

3.2.2.1 Excluded Papers

The final number of papers that were included for data analysis were 71 and 80 papers were excluded. Paper exclusion was caused by a number of reasons related to format, lack of peer review, lack of AUCDI focus, lack of focus on XP or Scrum, unavailability on-line, time constraints, redundancy and lack of quality. Each of those categories will be discussed in detail below:

Researchers Contacted	Contact Reason	Action
Dr.Paul Cairns	Unable to locate British HCI 2006 Proceedings on-line	Borrowed hard copy
Carol Barnum	Need to inquire on the presence of a peer review paper that discusses the reported study in the 90 min presentation at UPA [55]	She replied back with a detailed journal article [56] that were included and her UPA presentation was excluded [55] for lack of peer review
Zahid Hussain	Need to identify more comprehensive paper for what is published in [123, 125, 126, 127, 128, 285]	He replied back with a more comprehensive journal paper [129] that it was included and [123, 125, 126, 127, 128, 285] were excluded
Zahid Hussain	Need to ask about a paper with the detailed statistically analysed results on the AUCDI on-line survey that was mentioned in [130]	He sent out a comprehensive journal paper [132] that it was included and [130] was excluded
Jennifier Ferreira	Need to ask about her PhD thesis on AUCDI	She replied that she will send it after she finishs the corrections. She did not send the thesis until the time when I finished the SLR so it was not included in the search

Table 3.6: Efforts to Find Missing Studies

Researchers contacted	Reason	Result
Antti Nummiaho	Need to acquire full referencing details for [220] and whether the paper was peer reviewed	The reply was that the paper represents work related to postgraduate courses and is not peer reviewed so it was excluded
Heather Williams and Andrew Ferguson	Need to ask about the Agile method used in [283]	The reply was that the Agile method was a mix of XP and Scrum and the paper was included after passing the quality criteria
Sisira Adikari	Need to clarify some information regarding [3]. This information included the Agile method, full referencing details and whether the projects were student projects or industrial projects	The reply was that the Agile method is Scrum and she provided the full referencing details and that the projects were industrial projects and the paper was included after passing the quality criteria
Helen Sharp	Need to get the full referencing details for [261]	She replied back with full referencing details and the paper was included
Abbas Moallem	Need to get full referencing details for [205]	I got no reply and the paper was excluded for lack of peer review
Jukka Haikara	Need to clarify more details on Mobile-D [102] and whether it is a variant of XP thus a newly proposed Agile process	I got a failure email due to change of address and paper was excluded because it is not focused on either XP or Scrum
John Eklund and Ciaran Levingston	Need to get full referencing details for [71] and whether the paper was peer reviewed	Authors were contacted by email and due to lack of reply I checked with Dr. Paul Cairns who confirmed that the paper is not peer reviewed so it was excluded

Table 3.7: Efforts to Identify Unclear Information

Exclusion based on format

A total of 16 papers were excluded because of their format, two papers [77, 260] were excluded since they represented a panel, 7 papers [47, 54, 217, 230, 232, 234, 271] were excluded because they represented a tutorial, three papers [207, 208, 273] were excluded because they represented a workshop, one paper [212] was excluded because it represented an interview, one paper [104] was excluded because it represented a seminar, one paper [204] was excluded since it was a special interest group moderated session and one paper [55] was excluded because it represented a 90 min Presentation at UPA 2008.

Exclusion based on lack of peer review

A total of 14 papers [4, 7, 44, 58, 71, 104, 133, 197, 205, 220, 233, 274**?**] were excluded due to lack of peer review. Although [194] was excluded due to lack of peer review, almost the same content was later found in another paper [195] that was peer reviewed and as a result it was included.

Exclusion based on lack of focus on AUCDI

A total of 17 papers [27, 72, 74, 90, 93, 107, 108, 157, 179, 188, 189, 190, 223, 236, 242, 266, 281] were excluded due to their lack of focus on AUCDI.

Exclusion based on lack of focus on XP or Scrum

Two papers were excluded due to lack of focus on XP or Scrum. Haikara [102] paper was not focused on either Scrum or XP but rather on Mobile-D that is described in more details in [1]. Krohn et al. [168] paper was also excluded since it was focused on FDD rather than Scrum or XP.

Exclusion based on unavailability

One paper [253] was excluded due to its lack of availability on-line although it were peer reviewed and accepted by the Agile 2010 committee.

One paper [218] was excluded since it was not available freely on-line and it costs 149 dollars for purchase.

Two PhD theses were excluded based on lack of on-line availability. First, Zahid Hussain PhD thesis, however, papers [129, 131, 132] related to this PhD thesis were included in the SLR. Second, Jennifier Ferriera PhD thesis, however, a set of papers [84, 85, 86] related to her PhD work were included in the SLR.

Exclusion due to time constraints

Two PhD theses were excluded due to the time required to read them and the time constraints involved in doing the PhD did not permit going through them. Tiago Silva da Silva PhD thesis [52] and Jason Chong Lee PhD thesis [175]. However, Tiago Silva da Silva papers [51, 53] and Jason Chong Lee papers [176, 177, 178] produced from their PhD thesis were included in the SLR.

Excluded Redundant References

The main focus of the SLR was to study AUCDI challenges, strategies and success factors. Thus a decision was made to exclude all papers written by the same authors that narrate the same AUCDI techniques or detail the early phases of applying a particular AUCDI technique and include the most comprehensive paper. In case of inability to decide from the papers or in case of having doubts the paper author was contacted to confirm the inclusion decision. As a result 18 papers were excluded for redundancy, details on reasons behind the exclusion decision are provided below:

- Lee [174] paper was excluded since it is an extended abstract in CHI and another more detailed paper [177] was included in the SLR instead.
- Patton [227], Patton [229] papers were excluded since the same contents and further details were included in another paper that was included in the SLR [228].
- Obendorf et al. [222] paper describes XPnUE and details an example of using it in teaching an academic course and developing a calendar project in a postgraduate course. Same details were included in [221] but with further reporting on both the academic project and another industrial project. Thus it was decided to include [221] in the SLR since it was more comprehensive.
- Ungar [277] paper was excluded and a more comprehensive paper citeUngar-CHI-2008 was included in the SLR that reports the same approach.
- Beyer et al. [21] paper was excluded since it was not peer reviewed and paper content was repeated in another paper [20] published in XP Universe 2004 and that was the paper included in the SLR.
- Ferreira [79] paper was excluded since it was a position paper and more details were included in [84, 85].

- Hudson [119] paper was excluded since it summarizes Jeff Patton's tutorial at OOPSLA 2004, and does not report on using it and since Patton's work was already included so it was decided to exclude this paper.
- Kollmann et al. [166] paper was excluded for redundancy since the full Masters thesis [165] that represent the same work was included. Kollmann et al. [166] paper was read to ensure that no new topics were covered in the paper other than those covered in thesis.
- Ferreira [80] paper was excluded since it summarizes literature and points out direction for future research related to values endorsed by organisations in which developers and designers are embedded. This paper was excluded since the same information related to values were discussed in detail in [84] that was included in the SLR.
- Hellmann et al. [109] paper was excluded since it did not include any details on evaluation and was followed by Hosseini-Khayat et al. [116] that included further work and details on the same topic and this paper was included in the SLR.
- A set of papers [123, 125, 126, 127, 128, 285] related to the same proposed AUCDI technique were excluded after contacting the author, Zahid Hussain, for advice on the most comprehensive paper. He advised to include another paper [129].
- Hussain et al. [130] paper was excluded since it mentioned that a future paper would include further details on statistical analysis and after contacting the author, Zahid Hussain, in regards to that matter he sent a more recent and comprehensive journal paper [132] with full details on statistical analysis that was included in the SLR.

Excluded Based on Quality

AUCDI studies fall into a wide spectrum of expert opinion, theoretical, experience reports, and empirical papers. This resulted in having to depend on a variety of quality assessment guidelines, tools and checklist that suit this multitude of research study types. 6 papers were excluded due to lack of quality after being subjected to the study quality assessment criteria that were mentioned in section 3.1.3.

As mentioned in section 3.1.3 it was decided to evaluate the quality of papers via a 3 point scale that signifies a ("No Attempt", "Attempt", "Full Attempt") state rather than a dichotomous ("Yes" or "No") grading scale in order to catch any attempt for achieving quality. All papers that score

less than half the final score of quality criteria were excluded, i.e. research papers include 11 quality criteria so all papers that has a quality criteria less than 5.5 were excluded. Thus a number of papers [28, 60, 91, 115, 180, 284] were excluded due to quality.

It is important to note that the decision to exclude the last four papers [28, 60, 115, 284] due to quality was influenced by their length since those three papers were short papers so most of the assessed quality criteria were not covered due to space limitations.

8 Papers were not evaluated for quality [8, 20, 118, 195, 251] since they represented theoretical papers and the quality criteria did not apply to them. However, they offer AUCDI challenges, strategies and success factors that are worth inclusion in the SLR. In addition, one workshop were included and exempted from inclusion and exclusion criteria; the CHI 2008 workshop [273] was excluded from quality evaluation due to its format as a workshop paper. Kollmann [165], Rannikko [243] Masters theses were not evaluated for quality. It was decided to include this workshop paper and the two Masters theses due to their valuable contributions to AUCDI challenges, strategies or success factors.

Appendix B includes the quality assessment for the different research papers according to the research papers quality criteria mentioned in chapter 3 in section 3.1.3.1.

Appendix C includes the quality assessment for the different experience reports according to the experience reports quality criteria mentioned in chapter 3 in section 3.1.3.2.

Exceptions to inclusion criteria

Although section 3.1.2.3 discussed the inclusion and exclusion criteria and it stated that workshops will be excluded. Two exceptions were made to that decision since the CHI 2008 workshop [273] was included due to being a pivotal and highly referenced paper that included a comprehensive summary of the key challenges faced by UX practitioners in doing AUCDI. In addition, the BCS-HCI Group's 7th Educators Workshop [261] was included since it discusses customer collaboration in XP and details the results from a focus group on Agile development conference 2003 as well as the results of a workshop held in XP2003 conference.

3.2.3 Studies Classification

Table 3.8 shows the results of the second stage of paper classification after taking into consideration the criteria for inclusion and exclusion stated in section 3.1.2.3. As shown in the table that a total of 80 papers were excluded for various reasons related to format, lack of peer review, lack of AUCDI focus, Lack of focus on XP or Scrum, unavailability on-line, time constraints, redundancy and lack of quality.

Exclusion Reason	**Number**	**Percent**
Paper Format	16	20%
Lack of Peer Review	14	17.5%
Lack of Focus on AUCDI	17	21.25 %
Lack of Focus on XP or Scrum	2	2.5%
Unavailability On-line	4	5 %
Lack of Time	2	2.5 %
Redundant Content	18	22.5%
Lack of Quality	6	7.5%
Total	80	100%

Table 3.8: Second Classification-Excluded Papers

Table 3.9 shows that 71 papers represented the final number of papers included for data analysis. These papers ranged between research papers, experience reports and papers that were included although exempted from quality assessment.

Research Papers	**Experience Reports**	**Papers Included but Exempted from Quality Assessment**	**Final number of papers**
39	24	8	71

Table 3.9: Final number of papers

Studies by Year of Publication

Table 3.10 shows the classification of the different studies according to the publication year, as it can be perceived that 2007 and 2008 had the largest number of papers, 11 papers in 2007 and 13 papers in 2008.

Year	Papers	Number	Percent
2000	0	0	0%
2001	[45]	1	1.4%
2002	[228, 246]	2	2.8%
2003	[118, 156]	2	2.8%
2004	[9, 20, 36, 153, 261]	5	7%
2005	[23, 113, 114, 193, 203]	5	7%
2006	[39, 183, 195, 200]	4	5.6%
2007	[59, 62, 81, 82, 83, 177, 198, 199, 226, 272, 283]	11	15.5%
2008	[8, 32, 33, 78, 89, 165, 181, 184, 209, 221, 262, 273, 278]	13	18.3 %
2009	[3, 35, 56, 76, 131, 136, 178, 239, 254]	9	12.7 %
2010	[5, 84, 116, 172, 267, 275]	6	8.5 %
2011	[34, 41, 53, 85, 120, 176, 185, 243, 251]	9	12.7%
2012	[86, 129, 132, 206]	4	5.6%
Total	—	71	100%

Table 3.10: Studies by Year of Publication

Studies by Conference

Table 3.11 shows the list of different conferences that included papers on AUCDI.

Studies by Journal

Table 3.12 shows the list of different journals that included papers on AUCDI. As shown in table 3.12, there is a scarcity in publications in journals and all journals included only one occurrence of publication on AUCDI related topics.

Studies by Magazine

Table 3.13 shows the different magazines that included publications on AUCDI. As noted in table 3.13 that only three magazines included AUCDI related articles. Those magazines were interactions, IEEE software and communications of the ACM. Interactions included three articles which is highest number of articles on the topic among the other two magazines.

Studies by Book Chapter

Table 3.14 shows the AUCDI related publications that were included as book chapters. It can be noticed from table 3.14 that only two book chapters related to the topic were published.

Name	No.	%	References
Agile Conference	19	34%	[32, 33, 34, 53, 78, 81, 86, 89, 113, 136, 156, 177, 178, 181, 200, 203, 209, 262, 283]
CHI	6	10.7%	[35, 176, 183, 221, 273, 278]
XP	6	10.7%	[5, 39, 83, 84, 206, 275]
HCI	3	5.4%	[62, 198, 261]
XP/ Agile Universe	1	1.8%	[20]
ICSE	1	1.8%	[251]
GI Jahrestagung	1	1.8%	[36]
OOPSLA	1	1.8%	[228]
International Conference on Software Engineering Advances	1	1.8%	[185]
The International Conference on Contemporary Ergonomics	1	1.8%	[195]
Symposium of the Work group Human-Computer Interaction and Usability Engineering	2	3.6%	[130, 131]
International Conference on Advances in Computer-Human Interactions	1	1.8%	[129]
International Conference on Computer Design and Applications	1	1.8%	[267]
Symposium on Human Interface 2009 on Conference Universal Access in Human-Computer Interaction	1	1.8%	[76]
New Zealand Computer Science Research Student Conference	1	1.8%	[82]
Product Focused Software Process Improvement	1	1.8%	[153]
HCI 2009	1	1.8%	[3]
International Symposium on Intelligent Information Technology Application	1	1.8%	[239]
Australian Conference on Information Systems	1	1.8%	[226]
International Computer Software and Applications Conference	1	1.8%	[114]
International Conference on Human-Centred Software Engineering	1	1.8%	[120]
Participatory Design Conference	1	1.8%	[246]
International Conference on Universal Access in Human Computer Interaction	1	1.8%	[199]
International Conference on Information Technology Interfaces	1	1.8%	[238]
Irish HCI	1	1.8%	[254]
Total	56	100%	–

Table 3.11: Studies by Conference

Name of Journal	No.	Percent	References
Journal of Systems and Software	1	14.3%	[132]
IFIP Advances in Information and Communication Technology	1	14.3%	[172]
Information Age	1	14.3%	[45]
Journal of Usability Studies	1	14.3%	[272]
Software Practice and Experience	1	14.3%	[85]
The Code 4 Lib Journal	1	14.3%	[184]
Cutter IT Journal	1	14.3%	[118]
Total	7	100%	–

Table 3.12: Studies by Journal

Name of Magazine	No.	Percent	References
Interactions	3	75%	[9, 59, 193]
Communications of the ACM	1	25%	[41]
Total	4	100%	–

Table 3.13: Studies by Magazine

Name of Book	Number	Percent	References
Maturing Usability	1	50%	[8]
Human-Centred Software Engineering-Integrating Usability in the Software Development Life Cycle	1	50%	[23]
Total	2	100%	–

Table 3.14: Studies by Book Chapter

Studies by Masters/PhD

Two Masters theses [165, 243] were included. Up to our knowledge there is also four PhD theses that are focused on AUCDI. All of which were not included due to time constraints or lack of on-line availability but papers related to their PhD work were included in the SLR.

- Tiago Silva da Silva PhD thesis [52] was not included but papers [51, 53] related to his PhD thesis were included in the SLR.
- Jason Chong Lee PhD thesis [175] was not included but papers [176, 177, 178] related to his PhD thesis were included in the SLR.
- Zahid Hussain PhD thesis was not included due to time constraints and lack of on-line availability but papers [129, 131, 132] related to his PhD thesis were included in the SLR.
- Jennifier Ferriera PhD thesis was not included due to the reasons mentioned in table 3.6, however, a set of papers [84, 85, 86] related to her PhD work were included in the SLR.

Studies by Publication Channel and Occurrence

Table 3.15 shows the total number of papers published via different publication channels, as shown in the table that conference papers represent the majority of publications.

Publication Channel	Number	Percent
Conference	56	79%
Journal	7	10%
Magazine	4	6%
Masters Theses	2	3%
Book Chapter	2	3%
Total	71	100%

Table 3.15: Studies by Publication Channel and Occurrence

Figure 3.2 highlights the number of papers identified per publication channels

This section discusses the results of the qualitative analysis of the set of included primary studies. Studies are classified according to descriptive and content information.

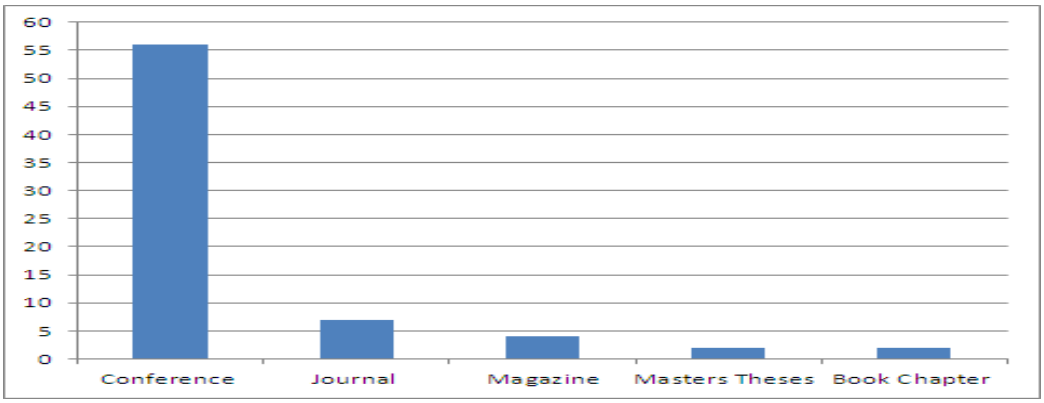

Figure 3.2: Studies by Publication Channel

3.2.4 Qualitative Analysis - Descriptive Classification

This section discusses the results of descriptive paper classifications. Papers were classified descriptively according to the type of Agile method and the research method used.

3.2.4.1 Studies by Type of Agile Method Used

The first type of descriptive paper classification depended on the classification of studies according to the Agile method used. The aim behind this classification is to identify whether some Agile methods suffer from insufficient integration research.

Table 3.16 shows the number and percent of Agile methods used in each of the primary studies. The selection criteria that were discussed in section 3.1.2.3 focused on two Agile methods XP and Scrum. However, some papers used the term Agile to refer to the Agile method used without explicating it.

Table 3.16 shows that there is almost an equivalent attention to Scrum (27 papers) and XP (28 papers) in regards to integration studies. Some primary studies had multiple classifications, i.e., included as both XP and Scrum, this usually occurs with empirical studies that conduct a set of interviews or surveys that span a number of projects, some of which utilise Scrum and the others utilise XP. Peixoto and da Silva [239] paper was specifically difficult to categorize since its abstract introduced it as representing some limitations of Agile methods and reporting on the results of

Agile Method	Number	References
Scrum	27	[3, 34, 35, 39, 56, 62, 76, 81, 84, 85, 86, 113, 131, 132, 136, 165, 172, 176, 178, 181, 183, 184, 206, 209, 262, 275, 278]
XP	28	[9, 20, 32, 36, 39, 62, 81, 82, 83, 113, 114, 129, 131, 132, 153, 165, 176, 177, 178, 193, 198, 199, 200, 221, 228, 246, 261, 272]
Agile	26	[5, 8, 23, 33, 34, 41, 45, 53, 59, 78, 89, 116, 120, 131, 156, 185, 193, 195, 203, 204, 239, 243, 251, 254, 267, 273]
Mix of XP and Scrum	2	[226, 283]

Table 3.16: Studies by Type of Agile Method Used

using Scrum in a specific project. However, after reading the paper it was discovered that it is focused on Agile and refers to another paper written by another author that discusses the results of a specific project using Scrum. So it was decided to classify this paper as an Agile paper rather than Scrum since that was the actual work done by the paper's authors.

3.2.4.2 Studies by Study Type

The second type of descriptive paper classification depended on the classification of studies according to the study type. The aim behind this classification is to identify whether the current state of research on AUCDI suffers any scarcity in particular study types and thus these study types need to be further exploited.

The type of included papers varied between theoretical, empirical, AUCDI tools, literature reviews, and systematic literature reviews papers. Theoretical papers reflect the authors' opinion on AUCDI challenges, success factors or strategies without subjecting their ideas to any form of validation. Empirical papers are those papers that investigate, describe, predict, and explicate a phenomena via utilising observations or experience based evidence [264]. The type of empirical research that was found in literature is: controlled experiments, surveys, and single or multiple case studies that were either implemented in academia or industry.

Table 3.17 shows that 17 of the papers were theoretical papers or Masters thesis that did not include empirical studies, for example, [243]. 63 pa-

pers were classified as empirical, one paper was classified as a literature review and one paper was classified as a SLR. Some papers were multiply classified. For example, Chamberlain et al. [39] paper was classified as a multiple case study that conducted both interviews, observations, and document analysis and as a result was classified in all three categories.

Three papers were focused on proposing tools for supporting AUCDI. Those papers are Hosseini-Khayat et al. [116] who presented "ActiveStory Enhanced", a tool for the creation and remote evaluation of low fidelity prototypes and Moreno and Yagie [206] described a tool for documenting usability into user stories. The third paper by Humayoun et al. [120] presented two automated tools: UEMan and TaMUiator. UEMan (user Evaluation Manager), is a tool for UCD management in integrated development environment and TaMUiator (Task Model-based Usability Evaluator), is a Java based tool that provides a set of APIs and interfaces for managing and automating task model based usability evaluation of the Integrated Development Environment (IDE) level.

Table 3.17 shows the research method used by the different included papers. It is important to note that although some papers claimed success of the proposed integration approach, however, it was decided to classify them as theoretical papers rather than case studies since they did not provide any details on the case studies, for example, [45, 114, 228].

3.2.4.3 Unclassified Papers via Descriptive Classification

Some papers were harder to classify via the above descriptive classification. Brown et al. [33] paper mentioned that the study covered eight companies, however, the paper reported the results of the analysis of one company only. Thus it was decided to categorize it as a single industrial case study rather than a multiple industrial case study. Moreover, it was not possible to classify Losada et al. [185] paper which proposes the following: First, InterMod, a new approach to improve software development by applying user centred and model driven development in an Agile manner. Second, a new integrated model, involved in a model driven process, to support the project requirements. Third, an Agile methodology organised as a series of iterations by means of user objectives as a novel method of promoting correct development. It was also not possible to classify Sharp et al. [261] paper since it reports the results of two investigations, a goldfish bowl focus group and a workshop, into the customer collaboration reality in XP.

Research Method	Type	No.	References
Theortical	–	17	[8, 20, 23, 45, 59, 62, 78, 114, 156, 195, 198, 204, 228, 239, 243, 251, 254, 273]
Empirical papers	Controlled Case Study or Experiment	1	[153]
	On-line Surveys	4	[56, 76, 132, 172]
	Single Case Study	23	[5, 32, 35, 36, 41, 129, 136, 176, 177, 183, 184, 199, 200, 203, 221, 226, 246, 262, 272, 275, 278, 283]
	Single Case Study-Observation and Interviews	3	[33, 85, 178]
	Single Case Study-Document Analysis or Artefact Analysis	1	[33]
	Multiple Case Study	5	[9, 113, 120, 181, 209]
	Multiple Case Study-Observations	6	[34, 39, 84, 86, 131, 165]
	Multiple Case Study-Interviews	12	[34, 39, 81, 82, 83, 84, 86, 89, 131, 165, 172, 193]
	Multiple Case Study-Document or Artefact Analysis	4	[34, 39, 84, 131]
Tools	–	3	[116, 120, 206]
Literature Review	–	1	[267]
Systematic Literature Review	–	1	[53]

Table 3.17: Studies by Study Type

3.2.5 Qualitative Analysis - Content Classification

This section discusses the results of content classification of papers. Papers were classified according to the integration approach and result.

3.2.5.1 Studies by Integration Approach

The first type of content paper classification depended on the classification of studies according to the integration approach. The aim behind this classification is to identify the current state of research in regards to the different integration approaches. Eight different categories were used to classify papers according to their integration approach as shown in table 3.18. Some papers are multiply classified.

Integration Approach	**No.**	**References**
Integrating Agile and UCD as Two Separate Processes	8	[8, 20, 23, 39, 45, 78, 114, 228]
Incorporating UCD Techniques into Agile	39	[9, 20, 23, 32, 35, 39, 41, 56, 59, 76, 81, 82, 89, 113, 120, 129, 131, 132, 165, 176, 177, 178, 184, 193, 195, 198, 199, 200, 203, 204, 209, 221, 243, 246, 254, 262, 273, 278, 283]
Adapting or Extending Agile Practices to Take UCD into Account	43	[3, 5, 8, 20, 23, 32, 35, 36, 39, 41, 59, 76, 81, 82, 83, 89, 113, 129, 131, 132, 136, 165, 176, 177, 178, 193, 195, 198, 199, 200, 203, 204, 209, 221, 243, 246, 254, 262, 272, 272, 273, 278, 283]
Adapting or Extending Organisational Practices to Suit AUCDI	17	[5, 23, 32, 35, 36, 59, 76, 89, 113, 129, 165, 181, 226, 243, 262, 273, 283]
Adapting or Extending UCD Techniques to Suit Agile Development Process	8	[3, 20, 23, 56, 129, 156, 243, 273]
Proposing a Tool Support for the Integration	3	[116, 120, 206]
Introducing New Team Roles	5	[5, 165, 221, 246, 262]
Investigating Developers and UCD Practitioners Engagement	5	[33, 34, 84, 85, 86]

Table 3.18: Studies by Integration Approach

3.2.5.2 Unclassified Papers for Integration Approach

Some papers did not fall under any of the above categories for example, Salah [251] paper proposed a maturity model for integrating Agile development processes and UCD. Sharp et al. [261] paper reported the results of two investigations, a goldfish bowl focus group and a workshop, into customer collaboration reality in XP. Larusdottir et al. [172] paper explored usability testing in Agile teams. Peixoto and da Silva [239] paper proposed a knowledge base representation of good HCI design practices and uses a semantic network for representing main HCI design concepts. Jokela and Abrahamsson [153] paper is a controlled case experiment that aimed to understand the extent to which XP guides the development of usable software. Duchting et al. [62] paper evaluated how Agile models consider usability engineering activities to guarantee usable software products creation. The user centrdness of Scrum and XP is analysed and the question of how potential gaps can be filled without losing the process agility is discussed.

3.2.5.3 Studies by Result

It was aimed to classify primary studies according to their results, i.e., whether they succeeded or failed in their attempt to integrate Agile development processes and UCD. The aim was to investigate failure integration studies in order to identify the causes of failure while successful integration studies were investigated in order to provide insight into successful integration practices or success factors.

Table 3.19 shows the classification of studies according to their results into failure or success.

Result	No.	References
Success	46	[3, 5, 9, 32, 33, 34, 35, 36, 39, 41, 76, 81, 82, 83, 84, 85, 86, 89, 113, 116, 120, 129, 131, 131, 132, 136, 165, 176, 177, 178, 181, 183, 184, 193, 199, 200, 203, 209, 221, 226, 246, 262, 272, 275, 278, 283]
Failure	1	[209]
Total	47	–

Table 3.19: Studies by Result

3.2.5.4 Unclassified Papers for Result

It was not possible to classify some papers as either success or failure. For example, theoretical papers [8, 20, 23, 45, 59, 62, 78, 114, 156, 195, 198, 228, 243, 251, 273], tool support papers [116, 120, 206], literature reviews [267] and systematic literature reviews [53], one of the on-line surveys [172] that was focused on usability testing methods rather than reporting success or failure, paper [204] that provided a a summary of key challenges faced by UX practitioners while practicing Agile UCD and proposed possible solutions, Sharp et al. [261] paper reported the results of a focus group and a workshop investigations regarding the reality of customer collaboration in XP, Peixoto and da Silva [239] paper that represented some limitations of Agile methods and reported on the results of utilising Scrum in a project in another paper by another author. Finally, Jokela and Abrahamsson [153] paper since it is a controlled case study that aimed to understand the extent to which XP guides usable software development and the majority of the details given focused on assessing the XP process from a usability perspective.

3.2.6 Agile and User Centred Design Integration Benefits

Academic researchers and industrial practitioners who attempted to integrate Agile and UCD reported several benefits achieved via the integration. The SLR revealed that AUCDI has positive impacts on the developed software, end users, development process, and the development team.

Software Developed

AUCDI result in improved software usability [3, 36, 36, 56, 130, 131, 184, 262], improved software quality [76, 130, 131, 184, 200, 228, 262] and improved UX quality [272].

End Users

AUCDI result in increased end user satisfaction with the developed software [130, 131, 132, 227, 228] which lead to increased end user acceptance [200].

Development Process

AUCDI adds value to the development process [131, 132]. The collaborative nature of Agile development processes guarantees early detection and

addressing of UX related problems [35, 184]. Memmel et al. [199] stated that the integration leads to reducing the risk of wrong design decisions via end users' involvement in pinpointing their needs and activities [199].

Development Team

AUCDI adds value to the development team [131, 132] via making it more effective [76] and facilitating knowledge transfer between developers and UCD practitioners that result in mutual appreciation [129, 131] since developers become more concerned about usability [56] and gain experience with UI design which result in delivering work that reflects polished UX [32]. AUCDI allow developers to gain more empathy and focus on end user [228, 231]. Moreover, the tight coupling of different expertise reportedly lead to improved motivation [129] and improve developers' productivity [262].

In addition, the integration lead to increased team confidence of the overall product [262] and what the software should do and why [184, 227, 228]. It also improves team morale [90, 126], sense of job satisfaction [76, 184].

Finally, AUCDI bridge the communication between stakeholders at the right times [32] and support planning and prioritization of activities [231].

3.2.7 Differences between Agile and User Centred Design

By pinpointing core convergence and divergence points between Agile development process and user centred design, this can lead to better integration that takes into consideration both the key differences that can hinder the integration as well as the key similarities that can facilitate it.

Section 3.2.7 and section 3.2.8 will discuss differences and commonalities between Agile and UCD.

Holistic View Versus Piecemeal View

Agile approaches, on one hand are incremental and iterative in nature since the software is developed and released in small pieces. Agile approaches do not generally support development of any kind of comprehensive and explicit overview of the software architecture. This is based on the view that requirements will change as users gain understanding of what the software can do for them. Thus effort spent on formalized requirements engineering could be better spent on developing code that can be evaluated by customers.

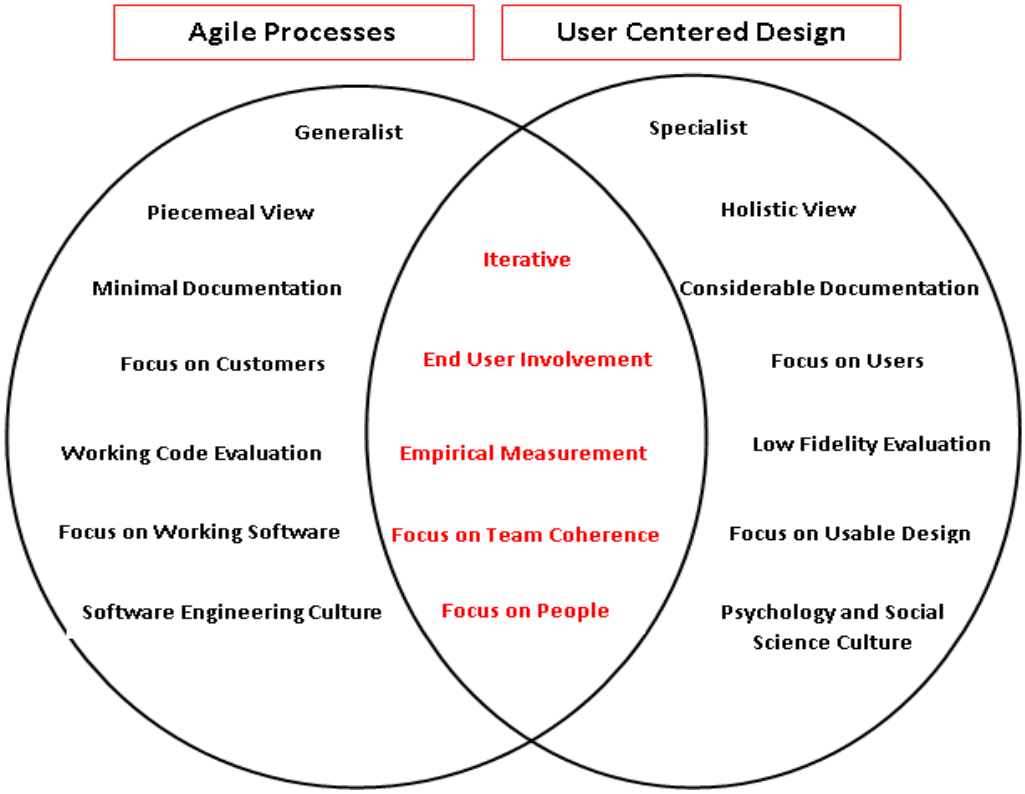

Figure 3.3: Commonalities and Differences between Agile and User Centred Design

User centred design on the other hand tends to be iterative but not incremental since UCD approaches typically start with developing ideas for different possible designs for the whole system and then refining it in later iterations. This aims to ensure the development of a consistent and holistic user interface [178]. Moreover, The iterative, fast and feedback oriented nature of Agile processes forces UCD specialist to modify their work practices into incremental design produced along the development of the software.

As a result Agile approaches try to avoid extensive upfront design phase [39, 221] however, this can lead to absence of a holistic vision at the beginning of Agile projects [195, 285], hindering consistent design [177, 195, 221] and developing user interfaces that are piecemeal, disjoint, and absence of a cohesive overall vision [178, 195, 221]. This almost inevitably results in an error prone, inefficient, and unfulfiling UX [195].

Focus on Customers Versus Focus on Users

The importance of user involvement has been recognised by human computer interaction academics and researchers for many years [39]. The Agile manifesto [14] requires the development teams to have an on site customer representatives in the development project, but it is not clear if these representatives are also end users. This lack of distinction between customers who commission software and the actual users of the software was criticised in Blomkvist [23], Cooper [48], Salah et al. [254]. Cooper [48] pointed out that there is an apparent mistaken assumption that customers are always the same as users. Moreover, Chamberlain et al. [39] stated that real users rarely take customer role.

This is a crucial issue since customer representatives are expected to make informed decisions about the scope and capabilities of the software being delivered [254], prioritize requirements, choose what will be included in or out of an iteration [261] and participate in defining and testing the software. However, the customer is usually either an expert or develops into an expert in the developed software via continuous work and presence with the development team. This makes it difficult for him to identify the different mental models, usability problems and concerns of different groups of users. UCD approaches clearly distinguish between users and other stakeholders such as customers. In UCD, actual users are pillars for providing information needed for UI design [193]. This was considered as common stumbling block for integrating Agile and UCD and a cause for Agile approaches to marginalize end-users and their roles in the development process in [23].

Focus on Working Software versus Focus on Usable Design

Agile approaches mainly focus on producing working software as a primary measure of progress, while usability approaches focus on producing a usable design. Nevertheless, usable design is not necessarily working software [79, 267] and working software is not necessarily usable. Moreover, code generated during sprints is often too unstable to conduct usability tests [59].

Evaluation Methods

UCD utilises low fidelity designs for evaluation purposes, while Agile approaches evaluates production ready code at the end of each iteration [193, 254].

Culture

The tight cooperation required in most Agile methods reveal the differences between engineers and UCD practitioners very quickly. Software engineers and human computer interaction practitioners come from different domains with different work cultures, backgrounds, attitudes, languages and approaches [254, 285]. Engineers are characterized by adopting a technical approach to software development based on understanding and using software concepts and software logic [254] while human computer interaction practitioners usually adopt cognitive view on the software development due to their psychological, empirical and social science background [254, 285]. Terms such as design, verification, validation, process and reliability may have different meanings to the different communities [254]. These differences pose problems for communication and collaboration between both parties and counter methods should be integrated into the development process.

Documentation

Agile approaches strive for minimal documentation [39] while UCD promoters express the need for certain documentation of design rationale in order to justify and record prior design decisions [193]. Sy [272] mentioned the need to document current designs and their expected delivery date, high level progress for late stage design chunks, usability test results, recommendations for working versions, task and user information and the user interface design to be implemented [272]. The lack of proper requirements documentation was reported to lead to confusion in regards to UX deliverable [35].

Generalists versus Specialists

UCD provides specialized user interface design skills, while, Agile approaches prefer generalists [193].

3.2.8 Commonalities between Agile and User Centred Design

The commonalities between Agile approaches and UCD methods can be perceived as convergence points that provides a solid foundation across the differences of focus, evaluation methods, culture, documentation, work cultures, backgrounds, attitudes, languages and approaches. These commonalities are as follows:

Focus on People

Both Agile and UCD are human centric development approaches [177]. UCD philosophy places the user at the centre of the development process while XP involves an on site customer representative in order to build software that appeals to users [89]. Agile approaches value face to face communication and coordination between team members [177]. Thus communication and collaboration is critical for development teams that involve usability engineers and software developers working in parallel in order to ensure that the software development track and the interface design track remain in sync [178].

End User Involvement

Both UCD and XP aim to effectively involve users in the software development process. This is achieved in XP via on site customer representative who is supposedly a potential user who contributes with iteration feedback. Yet there is a lack of details that help practitioners to pinpoint who those users are and how best to involve them before giving them such crucial responsibility [261]. In Scrum it is achieved via encouraging user presence in product evaluation during sprint review that occurs on a monthly basis [257]. UCD has a variety of techniques for enhancing user involvement, for example, participatory design, ethnographic methods and usability testing.

The question of how best to involve users represents a challenge to both UCD and XP. However, although there is considerable research in the UCD domain that proposes answers to this questions. Yet, this question is rarely tackled in the Agile domain and can be considered as largely unanswered

[261].

Iterative Development

One of UCD's founding principles is iterative design [39, 95, 177]. UCD proposes a design test modify cycle for the development of UIs [128, 195]. Iterative design tries to rectify problems encountered by users in usability testing [89].

Agile methods iteratively build working software. This reduces project risk via regular feedback, contributes to visibility of the project, enhances continuous improvement and allows for early achievement of business benefits. XP embraces constant and extensive testing and releasing of smaller working software [9, 23]. XP rely on iterative development and feedback as one of XP's values that is embodied in automated testing and code refactoring [39]. Iterations are usually cycles of one or two weeks [195].

Nevertheless, the iterative development in Agile and UCD differs. This difference is embodied in a number of issues: first, UCD's iterative design is more complex due to user involvement [128]. Second, the methods of refactoring and automated testing since automated testing is considered to be a development prerequisite. However, automated testing is useful in non usability related aspects since the presence of human being is required to evaluate the system [9]. Constantine and Lockwood [46] referred to automated user-interface testing as time consuming, labor intensive and extremely difficult except at the most elementary level.

However, UCD iterative interface development can be fitted into XP iterations since both Agile and UCD value evolutionary development [9, 128]. Explicit usability evaluations can complement automated functional tests and customer acceptance tests [23].

Empirical Measurement

Testing is integral to both Agile and UCD. Involving users in software development life cycle is a common aim to both UCD and XP. The question of how to effectively involve users remains as a challenge to practitioners in both Agile and UCD domain. Of the three UCD principles: early focus on users and tasks, empirical measurement and iterative design, the first and last are fundamental parts of XP [95, 261]. However, usability testing is totally ignored in XP [261]. Nevertheless, focus on users is not totally ignored in XP since the on site customer is supposedly a potential user who contributes with feedback on iterations. Yet there is a lack of details that help practitioners to pinpoint who those users are and how best to

involve them before giving them such crucial responsibility [261].

Focus on Team Coherence

The value of team coherence is emphasized by both UCD and Agile. The goal of the planning game is to bring the team together [15]. UCD approach also focus on bringing the team together by having one common focus on users throughout the development phases of the product [39].

3.3 Agile and User Centred Design Integration Challenges

This section is focused on reporting the results of the SLR research questions discussed in section 3.1.1.2 in regards to AUCDI challenges, practices and success factors. It is divided into a number of sub sections each of which deals with a particular AUCDI challenge and then lists the practices or success factors that have been reported in literature to tackle each challenge. These challenges fall into three main categories: UCD infrastructure, people, and process.

3.3.1 Lack of Allocated Time for Upfront Activities

Agile Methods discourages upfront planning activities since it strives to remain responsive to changing requirements [82, 178]. Moreover, Agile approaches focus on frequently producing deliverable solely in terms of functionality [183, 262]. This has resulted in lack of allocated time for software design planning activities [76, 165], performing user research [39, 59, 76, 165, 277] to discover the problems, work practices and work flows of end users [59, 76] and sketching out a coherent design [76, 165, 183, 277]. Incremental Agile development is translated into sliced or "feature by feature" development for design that can result in user interface that is disjoint, piecemeal and lacks a holistic, coherent, and overall structure and vision [5, 9, 165, 178, 183, 193, 195, 200, 204, 221, 262]. This lack of product's holistic design view can result in loss of shared vision due to immersion in details that lead to difficulties in prioritizing design decisions [204, 272]. This situation is also aggravated by the lack of documentation for earlier design rationale [193].

Practices and Success Factors

The common strategy that is used to deal with lack of allocated time for upfront activities in Agile teams is upfront design.

3.3.1.1 Upfront Design

Establishing a coherent design requires understanding users, context and customer goals [165]. Upfront design is a separate pre-development period that is used in Agile development projects for eliciting requirements, understanding users, user goals and context of use, using backlog items to create user stories and conducting UX design up front and ahead of developers in order to achieve a comprehensive system view [35, 39, 59, 89, 89, 113, 131, 132, 136, 165, 193, 193, 203, 209, 272, 283]. Upfront design can also be used by the development and quality assurance teams to work on back end features such as selecting development environment and system platforms [209] or features with low design cost and high development cost [203]. Upfront design is also referred to as an explore phase,"Cycle Zero" or "Sprint Zero" or "Iteration 0" or "Release 0".

Budwig et al. [35] reported that lack of time for upfront design led to increased work pressure and wasted effort for UX team in order to keep up with the pace of the development team, whereas utilisation of sprint 0 resulted in decreased churn and improved life work balance for UX team members [35]. Ferreira et al. [83] reported the positive effect of upfront design in mitigating poor design judgments, poor task prioritization, costly redesign problems, usability problems and inaccurate work estimates.

In regards to the duration of the upfront design or cycle zero, it varied; between two weeks [5, 32], four weeks [89], entire release [76] for very large scale or unprecedented projects or for large features that is impossible to chunk into sprints, or two iterations [283]. While Coatta and Rutter [41] declared that the duration of upfront design is dependent on reaching a good set of agreed upon user stories that include clear definitions of the associated work flows and use cases.

Techniques Used in Upfront Design

A number of techniques were reportedly used during upfront design: the design studio [277], personas [32, 82, 129, 165, 193, 262], extreme personas [129], contextual inquiry [20, 89, 193, 195, 203, 221, 272], low fidelity pro-

totypes [3, 9, 32, 41, 45, 89, 129, 165, 165, 195, 199, 200, 228, 262], project chartering sessions [283], surveys [184], focus groups [184], little design up front [3], and scenarios [9, 177, 221].

3.3.2 Difficulty of Modularization/ Chunking

Sy [272] defines design chunking as breaking design into cycle sized pieces called design chunks that incrementally add elements to the the collective design and design goals [272]. The incremental nature of Agile processes makes design chunking more critical and challenging [78, 203, 272]. This is due to a number of reasons: first, difficulty in determining the right chunk size and the right amount of interaction design work per iteration [275]. Second, difficulty of maintaining the ordering dependency between design chunks [272]. Third, difficulty in differentiating between user experience design activities that contribute to breadth or depth [113]. Fourth, interaction designers adopt a holistic view to interaction design and as a result it can be difficult for them to grasp and adopt design chunking both as an idea and as a work procedure [272]. Fifth, some complex designs require more than one cycle to complete [203].

All these difficulties lead to inability in estimating the time required for finishing UI related activities and consequently the required time to finish the UI especially in highly creative and artistic (User Experience Design) UED activities, complex features or unprecedented large scale projects [76, 113, 203, 209]. However, design chunking has a number of pros,for example, offering the UCD practitioners that ability to combine different usability investigations methods and to gather more data from less number of users [272].

Design chunking was tackled via a number of practices as follows:

Practices and Success Factors

A number of practices were utilised by Agile teams to deal with modularization/ chunking: having well defined design goals [272], using one release to chunk large or complex features [76, 209], chunking design into features [209], time boxing highly creative UX design activities [113] and postponing depth based UX activities to occur later in the development life cycle in order to develop both the functional feature and its related UX design activities in the same iteration [113].

3.3.3 Difficulty of Prioritizing UCD Activities

Inclusion and prioritization of usability or UX related activities into the different iterations or sprints is reported to be challenging as a result of developers' focus on accomplishing functionality features rather than usability or UX features [204, 262]. Moreover, Singh [262] reported that although usability tasks could be included on a backlog yet they usually do not get assigned enough priority to be included in the current sprint.

Practices and Success Factors

The difficulty of prioritizing UCD activities was handled via a number of techniques: assigning this responsibility to the designer or UCD practitioner [183, 204], having a separate UX product backlog [35], Uscrum [262], and having a separate UX Scrum team who is responsible for ensuring the prioritization of UCD activities [35].

3.3.4 Optimizing the Work Dynamics Between Developers and UCD Practitioners

Ferreira et al. [81] declared that Agile development process changed the relationship between developers and UI designers. Ferreira et al. [82] reported that the iterative nature of XP development process requires the continual involvement of interaction designers in product development and as a result impacted the relationship between interaction designers and developers. Moreover, McInerney and Maurer [193] reported that Agile development process has led the UI design to become a team effort and requires the UI designer to be on call to take part in discussions that are ad hoc in nature. Detweiler [59] reported that due to the highly compressed time scales and reliance on team self governance of Agile development processes. This requires more active involvement from UX managers to ensure the regular inclusion of UX activities in team based planning and scheduling.

However, the Agile principle "Working software is the primary measure of progress" can pose a challenge on the integration since it can introduce competing goals between usability and development- especially when they work in parallel. Moreover, the Agile principle "Simplicity–the art of maximizing the amount of work not done–is essential" can represent an additional challenge to AUCDI efforts since UI simplicity does not always align with implementation simplicity [178].

Agile development focuses on interaction and collaboration between people. The communication between developers and UCD practitioners in agile teams is embodied in the form of collaboration, cooperation, and coordination .

Communication

Communication is at the heart of all Agile teams and is embodied in a number of the principles of the Agile Manifesto [14] that states that "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation." and "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly." Ferreira et al. [82] emphasizes the importance of maintaining ongoing and continuous communication between the designers and developers to avoid the occurrence of delays and bottle necks in the development process.

Collaboration

Collaboration is defined as "a direct way for individuals to engage with others to achieve a joint purpose such as cooperating, or coordinating one's activities with others" [34]. Collaboration can be contrasted with simply interacting; since interacting represents communication with no joint purpose [34]. Furthermore, Ferreira et al. [85] stated that the interactions between the UX designers and Agile developers were "localized, contingent and purposeful". Collaborations are complex events that occur via artefacts, talk, gestures, electronic media, or combinations of these communication channels [34]. Interaction designers and software developers collaboration is critical to Agile projects' success. Coatta and Rutter [41] stated that Agile requires close team member collaboration and frequent face to face communication between stakeholders with considerably different technical vocabularies and backgrounds. As a result one of the Agile teams' biggest challenges is to ensure clear and effective communication.

Lee et al. [178] declared that communication and collaboration are critical for Agile teams where software developers and usability engineers work in parallel in order to ensure the synchronisation of software development track and interface design track.

Cooperation

Ferreira et al. [85] stated that designers and developers work is cooperative rather than collaborative. Ferreira et al. [85] stated that UX designers perceived cooperation with developers to involve passing on the UX designs

when the UX designers are satisfied they designed the best possible solutions.

Coordination

Communication and coordination across teams may be more heavily dependent on UX members to ensure compliance to accessibility legislation and UI style guides [59]. Ferreira et al. [85] pointed out that accomplishing UX designers' work relied on back and forth switching between designers tasks and tasks involving others with explicit and implicit articulation work to coordinate their switching.

A number of practices were utilised in order to optimize the work dynamics between developers and UCD practitioners. These practices will be discussed in the following subsections

3.3.4.1 Sharing an Understanding of Users

Understanding the end user characteristics and needs is essential to design good UX [165]. Miller [203] emphasized that investing time to ensure that the entire team understands and agrees on the target audience results in ease in the collection and utilisation of customer input throughout the development process. It also allowed the UCD practitioners to stay true to their vision and enabled them to make decisions on feature sets and design trajectories.

3.3.4.2 Sharing an Understanding of the Design Vision

Ambler [8] declared that the collaboration between software engineers and usability specialists should be supported via facilitating the communication of the rationale and intent of design. Kollmann [165] declared that the best UX vision is useless unless communicated to the development team. Thus UCD practitioners should effectively share the design vision via communicating it to the development team. This visibility of design vision minimizes rework and illuminates integration issues early on [277] and allows team members to develop and be aware of the system's key design goals in order to ease decision making in case of competing concerns. Setting the shared design goals allows team members to share a common understanding of important system aspects from the customer's perspective. Prioritized goals also allow the team to prioritize fixes [178].

Memmel et al. [199] encourages earlier externalization of design vision to stakeholders in order to make the development of usable software more effective, achieve better collaboration, and produce better software faster. All stakeholders should be able to collaboratively discuss the look and feel of the UI from the very beginning, visually express and share their ideas and cross check the outcome with their requirements.

Lee and McCrickard [177] stated that cooperation and communication between customers, designers, users and other stakeholders with different backgrounds and expertise can be supported via a shared design representation for high and low level interaction design views. This results in allowing different stakeholder groups with different backgrounds to understand and provide feedback on the design. It also explicates the interplay of Agile development and usability, and helps in having more focused planning meetings via clarifying key design concerns and decisions to stakeholders.

Ferreira et al. [81] noted that the daily communication of UI issues by the designer to the development team resulted in mutual understanding of how the UI design worked and how it was altered and allowed developers to have s significant input into the UI design.

Practices and Success Factors

Sharing an understanding of the design vision was achieved via a number of techniques: the design studio [277], face to face communication [165], engaging developers in multiple design options [193, 203], developers taking part in UI specifications [5], sharing design artefacts and prototypes [33, 34, 76, 177, 183], utilising information radiators [165], and extreme scenario based design [178].

3.3.4.3 Acceptance of UCD Practitioners by Development Team

Najafi and Toyoshiba [209] discussed the importance of successful integration of the UX team with functional teams since the lack of participation and continuous feedback from them in the development process can result in incorrect interpretation and implementation of designs by the engineering teams. Successful integration of the UX team requires full collaboration and cooperation with all cross functional team members [209].

Practices and Success Factors

Overcoming the barrier of accepting UCD specialists in Agile teams can be achieved via a number of methods: clear role definition [89, 176], UCD practitioners adopting the Agile mindset in order to ensure that simplicity and incremental development are not counterproductive to design [165, 178], mutual awareness between developers and UCD practitioners on each other's work mechanics, technical limitations, operational challenges and positive impact [8, 129, 277], work dependencies, timing, communication and information sharing mechanisms [86, 176], and respect and trust between developers and UCD practitioners [82].

3.3.4.4 Coordinating the Activities of UCD Practitioners and Agile Developers

Hodgetts [113] stated that the challenges of integrating Agile and UX occur due to the need to complete all of the code producing activities, from architecture through design, programming and testing within a single iteration [113]. This implies the need for continuous communication and coordination between developers and UCD practitioners. Ferreira et al. [85] pointed out that accomplishing UX designers' work relied on back and forth switching between designers tasks and tasks involving others with explicit and implicit articulation work to coordinate their switching.

Practices and Success Factors

Coordinating the activities of UCD practitioners and Agile developers can occur via utilising the parallel/ dual or staggered track. Parallel tracks [203] involves performing the implementation and design as two equal and highly interrelated tracks. The parallel track is organised around a number of cycles: cycle 1, involves work by designers on designing interfaces for features to be implemented by developers in cycle 2. Then low or high fidelity prototypes are built to test the design and design problems revealed during the usability tests are corrected, fixed in the prototypes and retested. This cycle continues until the designs achieve their design goals. Cycle 1 is utilised by developers in working on features with high development costs and little user interface [203]. Parallel tracks are especially beneficial when features are too big to be designed, usability tested, iterated, validated, built, translated and documented in one sprint [76].

The parallel approach was utilised by many researches and practitioners [41, 76, 89, 136, 203, 272] and [209] who referred to it as the staggered

development approach. Dual track offers a number of advantages for both interaction designers and developers. Interaction designers are always designing for the next iteration thus no time is wasted in creating unusable designs and they receive timely feedback. Whereas developers are able to maximize coding time since idle time waiting for designers to complete paper prototypes and usability tests is minimized [203].

3.3.4.5 Synchronizing UCD Practitioners and Developers Efforts

Design drift is the occurrence of a difference between the implemented system from the initial design as as a result of combining the efforts of developers and UCD practitioners. Synchronization allows the parallel usability and development efforts to proceed relatively smoothly. Detweiler [59] reported that UI consistency may be undermined as independently empowered teams evolve code in parallel, without coordinating their work. As a result, synchronization points are needed to allow for close collaboration that will keep the information flowing between all parties involved in the project. UX practitioners reported that the lack of communicating frequent changes caused a lot of confusion and required an immense effort from the UX team to handle frequent changes in addition to struggling to remain on track with the development team schedule. This reportedly resulted in negative life work balance for the UX team [35].

Brown et al. [34] observed that 92% of the designer developer collaborative events were related to realigning individual understandings or individual work to meet project and product aims. This realignment occurred via various forms, for example, scheduled, impromptu collaborations or informal chats [34].

Practices and Success Factors

A number of techniques were used by Agile teams to synchronize the activities of UCD practitioners and developers: attendance of UX team in daily Scrums [34, 35, 209], daily communication of UX designers to clarify design and inform the developers about additions or changes required for the UI [5, 82, 203], and increasing the visibility of UX team's work [32, 59, 176] via standup meetings [32, 35, 203].

3.3.4.6 Sharing Usability Testing Results and Recommendations with Development Teams

Illmensee and Muff [136] reported the positive impact of sharing research findings with stakeholders, customers, and management during weekly demonstrations. The regular reporting of weekly research sessions results allowed the development team to discuss utility and usability along with functionality and features, provided the team with a greater sense of accomplishment, demonstrated to stakeholders a collective concern for user satisfaction and quality, and made the development team accustomed to refining functionality according to usability sessions data rather than depending only on team members' opinions [136]. Miller [203] mentioned that interaction designers presented usability test results to the development team.

Practices and Success Factors

A number of practices were used to share user research and usability testing results and recommendations with team, for example, involvement of developers in data synthesis [59, 136], attendance of user testing sessions by development team and product management [129, 209], and making a highlight video for usability tests and showing it to the development team in order to support a user centred mind set in the development team [129].

3.3.4.7 Overspecialization and Existence of a Separate UED Organisation

Hodgetts [113], Lee and McCrickard [177] discussed the impact of increased specialization within Agile team members. Increased specialization results in less team cohesiveness [177], finer grained tasks and more hand off points between team members in order to complete a feature. Finer grained tasks impose difficulties in goals coordination and completion per iteration since they result in larger and more complex iteration plans. Whereas, the increased hand off points create more opportunities of misunderstanding and missed work [113]. Over specialization lead to pretask assignment which hinders the teams's velocity and ability to rapidly and flexibly respond to change [113, 177] and requires additional planning time that results in increasing the length of each iteration's planning activities [113]. Hodgetts [113] reported that the existence of a separate UED organisation imposes difficulties when team composition needs adjustment. This is due to the fact that rigid management and organisational hierarchy

leads to delay until acquiring management approval to any changes in personnel assignments and adjustments. This also leads to jeporadising the self organising teams which is characteristic to Agile projects.

Practices and Success Factors

A number of practices were used to deal with overspecialization; members with separate expertise or focus areas should strive to understand each other's specialties in order to avoid wasted work and misunderstandings [177]. Ferreira et al. [85] recommended enabling teams to self organise itself when combining Agile development with UX design. Ambler [8] stated that Agile practitioners have tightened the feedback loop on software projects, and consequently reduced cost and risk via abandoning the concept of teams of specialists and forming teams of generalists. Although UEX practitioners represent a critical skill set to the development team, they need to acquire a wider skills range to become highly effective.

3.3.5 Performing Usability Testing

Usability testing involves measuring typical users' performance on carefully prepared system tasks while watching and recording users performance and logging their software interactions [129, 241]. This observational data is used to calculate performance times, compute time of task completion, identify number and type of errors and explain reasons behind users actions [129, 241]. Nielsen [214] considered usability tests as the most fundamental UX evaluation method.

A number of sub challenges were related to usability testing within an Agile context: method of usability testing, timing of conducting usability testing/scheduling usability testing, accessing users for usability testing, finding time to analyse usability data, and high cost of running usability sessions. Further details on those sub challenges is provided below:

3.3.5.1 Method of Usability Testing

Agile time boxed nature poses challenges on scheduling and conducting usability tests due to the difficulty of scheduling usability tests to evaluate and test prototypes and working builds with representative end users [56, 59, 76, 78, 177]. Dayton and Barnum [56] reported that the Scrum process provides little time to include usability testing into a sprint and

unless a team member successfully argues to priortise usability testing, it is unlikely to be included because the always overriding, urgent goal is to develop working features at the end of each sprint. As a result, some Agile teams resolve to either peer test or do without usability, thus jeopardizing the quality of design, for example McInerney and Maurer [193] reported that the UCD specialist sometimes conducted informal usability testing with programmers working nearby.

Practices and Success Factors

The effect of tight Agile time lines on scheduling and conducting usability testing was reportedly overcome via a number of methods, for example, preparation for user research [136], utilising discount usability engineering techniques including: heuristic evaluation [89, 198] and RITE [56, 76], using low fidelity prototypes to conduct usability tests including: paper prototypes [39, 89, 131, 200], screen shots [131], and power point presentations [89, 131], conducting remote usability testing [59], utilising claims [177], and using high fidelity prototypes [41, 199] .

3.3.5.2 Timing of Conducting Usability Testing/Scheduling Usability Testing

Ferreira [78] reported that scheduling interaction design evaluations with Agile development iterations was considered as a challenge due to lack of clarity in regards to timing of evaluations as part of the iterative structure of the Agile development process. Kane [156] stated that conducting usability testing at the end of the Agile development process could lead to insufficient resources and time to respond to emerging usability issues while, if usability tests were done early in the development process this could lead to introducing usability defects in later iterations. Moreover, if usability tests were carried out as frequently as feature acceptance tests this could lead to massive budget increases. Furthermore, Detweiler [59] stated that code generated during sprints is often too unstable to be subjected to usability tests even if it was scheduled.

Practices and Success Factors

Ferreira et al. [81] perceived the completion of iterations and releases as opportunities to frequently test the software usability and declared that usability testing fits well with acceptance testing. Some researchers suggested fitting usability testing in the context of other Agile development

tests, for example, Carbon et al. [36], Ferreira [78], Hussain et al. [129] suggested acceptance testing sessions (in the case of XP) and Kane [156] suggested demonstration sessions (in the case of Scrum) could serve as opportunities for usability feedback on the implemented interaction design. Albisetti [5] introduced a mandatory UI reviews as a gate keeping tool, where two sign offs were set one for code and one for UI.

3.3.5.3 Accessing Users for Usability Testing

Detweiler [59], Ferreira [78], Hodgetts [113] reported that the compressed Agile time scale posed difficulties in organising access to the right people at the right time for usability testing. This is due to the need to plan user involvement, schedule appointments with studies subjects sufficiently in advance and thus may not fit with the Agile development schedule since it may require lead times of weeks. Larusdottir et al. [172] reported the results of a a survey conducted in Finland on Scrum teams to investigate how usability testing is conducted and the study revealed the lack of user commitment and willingness to take part in usability testing due to busy work schedules and lack of desire to be involved in software development.

Practices and Success Factors

A number of practices were utilised to maintain the ability to access users for usability testing in Agile teams, for example, planning in advance for user inclusion [165], utilising an existing user pool to act as development partners or design partners to conduct usability testing [3, 76, 283], using user recruiting firms to frequently schedule for usability sessions [136], conducting remote usability testing [59, 113], and collaborative (peer review) UI inspections [198] via designers, developers, end users, graphics designers and usability specialists.

3.3.5.4 Finding Time to Analyse Usability Data

Ferreira [78] reported that interaction design requires an implement-evaluate cycles that was difficult to keep up with the fast paced coding iterations. As a result, feedback on interaction design does not arrive in time to be integrated into subsequent development. Moreover, Illmensee and Muff [136], Lee and McCrickard [177] reported that analysing usability data is time consuming.

Practices and Success Factors

Agile teams tackled the problem of tight time for data analysis via utilising lightweight data analysis methods [136] and conducting smaller tests [129].

3.3.5.5 Shorter Time to Integrate Usability Testing Feedback on Design (Iterate Design)

The Agile tight time lines allow little time to integrate usability testing feedback into subsequent development cycles. Hussain et al. [129] declared that the reporting period for usability testing was too long and by then many changes have already occurred in the application and as a result many recommendations were obsolete. Moreover, Detweiler [59], Miller and Sy [204] reported lack of time to respond to results of usability evaluations and user feedback.

Practices and Success Factors

The challenge of shorter time to iterate design with user feedback was handled via dedicating cycles for working on user feedback and incorporating it into the development life cycle [193] and utilising the UX practitioner to act as an Agile customer in order to validate designs that are passed to developers to implement, participate in cycle planning and ensure incorporation of user feedback into the development life cycle [204].

3.3.5.6 High Cost of Running Usability Sessions

Larusdottir et al. [172] conducted a study that investigated usability testing practices in software development teams in Iceland using Scrum. The interview investigated causes for not conducting usability testing; the results revealed that lack of training was the reason in 20 % of the cases and lack of budget in 15 % of the cases and lack of time in 35% of the cases. Moreover, Illmensee and Muff [136] stated that the high logistics and cost of securing usability testing facilities involves: formal labs with two way mirrors, observation rooms, and recording equipment were prohibitive to Agile development team.

Practices and Success Factors

The high cost of running usability sessions was dealt with via utilising conference rooms instead of usability labs to conduct usability tests [136], conducting heuristic evaluations [129] and conducting smaller tests [129].

3.3.6 Absence of UCD Practitioner

Agile teams structure varies, some teams have a dedicated UCD practitioner whereas others do not. This can negatively impact the quality of product usability or UX. Brown et al. [34] declared that the interface designer place on Agile teams is ill defined. Blomkvist [23] discussed the scarcity of a specialized role in Agile teams with the skills and responsibility to coordinate the interaction design work and as a result usability and design lies in the hands of users or customers or developers [23]. Thus customers or users are held responsible to define the product features they want, prioritize them, and communicate them to developers [23]. Ferreira et al. [82] stated that the presence of interaction designers on XP teams is essential for achieving good interaction design whereas having developers doing interaction design is not ideal. However, Ferreira et al. [82] pointed out that although it is important to have a dedicated UCD specialist role in Agile teams. However, what is most important is for the UCD specialist to adopt the Agile mindset and adapt to the development process to work efficiently since UCD specialists are not used to the iterative, fast and feedback oriented nature of Agile methodologies and as a result they are forced to change into incremental work practices rather than big upfront design.

Hussain et al. [132] reported the results of an AUCDI on line survey that had 92 respondents who were UCD professionals and Agile developers. 52 % preferred having a dedicated UCD professional, 4 % indicated that this role is performed by a developer having interest in HCI/usability and 36 % gave no answer [132]. One respondents indicated that "you cannot get good UI with a developer doing it, they will always have a conflict of interest focusing on what is easier to build rather than what is the best user experience". Another respondent indicated that "having a dedicated UCD person is always better, very few developers I have seen with an interest in UCD are actually able to focus on the user rather than just 'cool design' or 'what is easier to develop'" [132]. Some respondents also commented for either situation: "Either can work: it depends on the skills of the person, it depends on the skill of the person, knowledge of the domain and

interpersonal team dynamics".

Practices and Success Factors

Hussain et al. [132] reported the results of an AUCDI on-line survey that had 92 respondents who were UCD professionals and Agile developers. 75% of the respondents believed that developers can pickup HCI skills by pairing with a UCD professional, 66% mentioned that this can be achieved via training [132]. Albisetti [5] reported choosing a number of developers who had an interest in UCD and mentoring them in performing UI reviews for other developers [5]. This resulted in better developers' productivity, less workload for the UCD practitioner, more experience gained by developers with UI reviews that resulted in improved work quality [5]. Moreover, Fox et al. [89] reported that lack of UCD specialist allows the developer to act as UCD specialist and as a result can immediately deal with any usability related issue instead of losing time by passing it to another team.

3.3.7 UCD Practitioner Workload

Although large organisations can afford a dedicated UCD practitioner per team or few teams which leads to ease in resource allocation and design consistency. Nevertheless, in smaller organisations UCD practitioners are usually either shared among a number of teams or work as part timers. Due to the specific nature of Agile process this can lead to an extra burden on UCD practitioners. Federoff and Courage [76], Leszek and Courage [181] reported the results of a survey that was conducted after 30 teams transitioned to Agile. Only 30% of the development services (DS) team that included members from documentation, usability and UX design believed that their teams are more effective after transitioning to Agile [76, 181]. The primary reasons behind that were that they were assigned to too many teams, attended lengthy a lot of time in meetings, and the tighter Agile time lines did not provide them with enough time to finish their work [76, 181]. Moreover, the frequent context switching resulted in a severe productivity loss for DS team members. So mainly the reasons were attributed to lack of resources since having the development services team members shared among a number of teams was no longer effective [76, 181].

Practices and Success Factors

A number of practices can be deployed for dealing with the increased load of UCD practitioners, for example, conducting mentoring process to developers so as they can perform the role of UCD practitioner [5], office hours [76, 181], decreasing UCD practitioner workload [76, 181] and distributing UCD practitioner workload on a UCD researcher and UCD a prototyper [283].

3.3.8 Lack of Skills (Competence)

Achieving Agile and UCD integration is dependent on several factors including the existence of usability awareness and skills in product owners, Scrum masters, Agile coaches, developers, customers and users. Blomkvist [23] pointed out the deficient focus of Agile processes on competencies essential for the software development projects and that although the role of customers, business people and developers is well described and filled in Agile teams yet the role of usability practitioners and interaction designers is largely overlooked due to lack of awareness among Agile practitioners on usability importance [23]. This lack of professional and skilled UCD staff heavily impacts the design and evaluation of usability in particular and producing usable products in general [23]. Moreno and Yagie [206] pointed out that customers, users and Agile developers are not usability experts and as a result a clear procedure is needed to achieve usability or else it will be dependent on customer and/or developer intuition. Moreover, Singh [262] discussed product owners' lack of UX design skills due to their focus on marketing and sales issues rather than usability concerns.

Practices and Success Factors

The challenge of lack of skills can be tackled via a number of practices including Uscrum in which two product owners are recruited one responsible for functionality and the other responsible for usability and UX [262], tool support in which usability information from different sources is recorded and presented or documented as part of user stories to enable proper estimation and implementation [206], education of UX integration in university courses [114], and developing lightweight style guides in order to convey best practices of common feel and look across a wide range of applications [156, 198].

3.3.9 Lack of Tools

Coatta and Rutter [41] claimed that the most crucial problem that faces the integration is the deficiencies of development tools since they do not provide sufficient separation between business logic and user interfaces. This leads to inability to modify user interfaces from top to bottom without impacting the underlying code [41]. Moreover, it makes developers more reluctant and less flexible to agree on any modifications to user interfaces suggested by UCD practitioners and thus places an additional burden on UCD practitioners to convince developers with user interface modifications and justify it via quantitative and qualitative usability testing results [41]. Moreover, Coatta and Rutter [41] stated that the different tools used by developers and UCD practitioners can contribute to communication problems. Moreno and Yagie [206] declared that tools could be the solution to the lack of usability expertise of customers, users and Agile developers.

Practices and Success Factors

Moreno and Yagie [206] created an open source tool that allow developers with limited usability knowledge or teams that do not have a dedicated UCD practitioner to create usable systems. This occurs via capturing usability meta knowledge related to the inclusion of particular usability mechanisms in a software system in order to enable proper estimation and implementation. Moreno and Yagie [206] implemented three methods for handling usability issues in Agile teams: first, representing usability requirements in the form of usability stories. Second, the tool transfered actions that are derived from usability constraints into additions or modifications of tasks in existing user stories. Third, in case of the need to include some specific usability related actions that modify the operating environment, this was dealt with via addition or modification of acceptance criteria.

3.3.10 Lack of Documentation

Kollmann [165] stated that although waterfall projects perceive documentation as a communication tool, however Agile approaches strive to achieve minimal documentation. Moreover, interaction with people reduces documentation and thus Agile team members perceive documentation as insufficient for communicating interactive behaviour [165]. However, doc-

umentation is crucial for estimation and implementation efforts and for properly integrating Agile and UCD. Furthermore, the lack of proper requirements documentation was reported to lead to confusion in regards to UX deliverable [35]. A variety of integration related issues need to be documented including: first, design rationale in order to justify and record prior design decisions [193]. Second, the source of requirements whether they are customers, users, developers, usability experts or usability elicitation guidelines because this can affect the decision of creating new user stories or modifying existing ones since this could have an impact on the workload associated with the respective user stories, and consequently on the sprint plan [206]. Third, Sy [272] mentioned the need to document current designs and their expected delivery date, usability test results, recommendations for working versions, high level progress for late stage design chunks, task and user information, and the UI design to be implemented.

Practices and Success Factors

A number of practices can be used to deal with the challenge of lack of documentation: documenting via Wikis [32, 221, 272], documenting via webpages [283], use cases [59], scenarios [221], personas [165], sketches [165], wire frames [165], prototypes [165], design patterns [193, 198], information radiators [165], tool support [206] and issue cards [272].

3.3.11 Non Colocation of the Development Team and the User Experience Team / Distributed Teams

McInerney and Maurer [193] reported that one of the possible impacts of the Agile process on UI design is that UI design became more of a team effort and the UI designer needs to be "on call" to participate in discussions that are ad-hoc in nature. Moreover, Williams and Ferguson [283] stated that colocation is perceived as best practice by many practitioners since it simplifies collaboration, allows for continuous communication, negotiation, knowledge sharing and enables instant decision making between developers and designers [275]. Fox et al. [89] stated that in case of colocation of UCD practitioners and developers the exchange of design is constant and ongoing. Hussain et al. [132] also stated that colocation makes it easier to influence the design as it progresses and helps the UX person become integrated within the team since he/she is available to answer questions and address issues that come up during the iteration. Colocation allows developers to be aware of designers work and spot design areas

that could cause problems in the implementation [275]. Ferreira et al. [84] stated that those who support the colocation promote favorable environments where stakeholders interact continuously to achieve work progress.

Lee et al. [176] identified three challenges for integrating Agile and UCD in a distributed environment these include: synchronizing distributed development and usability efforts, promoting team members communication, and effectively supporting the sharing of document and artifacts among team members who are physically separated. Fox et al. [89] reported that in the case of non colocation of UCD practitioners and developers the exchange of design got delayed to the end of each iteration. Furthermore, Sy and Miller [273] suggested that non colocation of team has a number of implications, for example, introducing time and language barriers, difficulty of communication [5, 176, 273], lack of sense of team , and lack of trust [273]. In addition, Budwig et al. [35] reported difficulties faced by the UX practitioners in coordinating with off shore development teams and as a result development and UX teams are collocated to decrease efforts of coordinating the work of remote teams [35]. Moreover, Najafi and Toyoshiba [209] reported that the geographical separation of UX team from the engineering team led to the exclusion of the UX teams from release planning, sprints and Scrum and lack of knowledge of UX team on implemented features per sprint, inability to conduct user research or testing of design detailed specifications and difficulty in understanding and agreeing on opportunities and constraints in terms of design options and understanding bigger picture.

In spite of all the acknowledged benefits of colocation Lievesley and Yee [183] refused to collocate interaction designers with the development team to adopt the culture of Agile development. This was due to a number of issues, for example, the need of interaction designers at the initial iterations to exert extensive mental effort to make sense of and synthesize diverse interests, information and influences. This would have not been possible to accomplish in an unfamiliar and tension laden environment with tactical and operational noise that could impede the designers' ability to envision the completed software product and position it in relation to the end user's requirements. Thus the design team worked independently of the development team in early creative stages, but a rigourous communication regime was abide with.

Nevertheless, Ferreira et al. [85], Ferreira et al. [86] research suggested that the non colocation of UX designers and Agile developers can be attributed to distinct work group cultures and organisational policy that shaped their

cooperation views. Ferreira et al. [86] stated that integrating Agile with UX design is shaped via a complex interplay of organisational and team commitments that determines how responsibilities are divided between developers and designers, establish each group work rhythms, and place different values on different roles' contributions.

Practices and Success Factors

The challenge of the non colocation of the development team and the UX team was dealt with via telecommunication tools [129, 273] that is utilised in chatting [129], instant messaging [176], emails [176, 204], phone or video conferencing [129], on line portal or discussion boards for asynchronous communication [176], collocating teams to attend cycle planning meetings [204, 273] and allowing distributed teams to attend daily meetings via phone conferencing and screen sharing software [176], introducing an information sharing process [176], and sharing design vision [183].

3.3.12 Maintaining Communication between the Customer and the Development Team

Agile approaches require development teams to include customer representatives who are expected to make informed decisions about the value of the software being delivered [14]. The customer in XP teams is expected to be integral to the development team and to perform a number of tasks, for example, generating requirements and acceptance tests, discussing user stories, answering developers' queries, setting product priorities, facilitating emergent requirements, and providing feedback on iterations [15]. The customer should be collocated with the developers; would be a potential system user, and would be collaborative, representative, authorised, committed and knowledgeable [24]. Detweiler [59] stated that contrary to recommendations of close partnering and frequent feedback with customers, many Agile projects engage customers only occasionally. Moreover, Rittenbruch et al. [246] pointed out that constant customer exposure to the development process can lead to identification with development problems and consequently losing focus on user issues. Sharp et al. [261] pointed out that ever since the Agile Manifesto was articulated efforts were exerted by XP practitioners to devise practices and methods to deal with the gap that exist between the reality and the ideal of customers in XP teams.

Customers have a significant role in successful integration of Agile and

UCD, Miller and Sy [204] reported that having a weak Agile customer can result in lack of end user participation and leads to making non informed decisions. Hussain et al. [129] suggested that proper customer and UCD practitioner coordination is necessary for the inclusion of the UX process in the development process. Broschinsky and Baker [32] also suggested that it is the customer's responsibility to make up to date usability tests reports visible to the developers. He also declared that since the customer is the one creating and prioritizing user stories it is his duty to think about and include UX aspects in his user stories. Moreover, Hussain et al. [129] suggested that an experienced on site XP customer can fill in the technical gap between UCD practitioners and developers.

Practices and Success Factors

A number of practices were utilised for maintaining the communication between the customer and the development team, for example, trust between developers and customers [261], customers understanding of both the world of customers as well as developers and understanding what is relevant to the business [261], planning ahead for UI design and usability testing in order to help the customer in iteration planning [129], maintaining a common vision between customers and developers in regards to what needs to be accomplished [261], continuous communication of usability engineering aspects to customer [114], contextual inquiry [32], personas [32], perception of customer as a "bridge" between the developer and the customer world [261].

3.3.13 Coordination of UX Team With Business Units

External dependencies slow down the development process [184]. Miller and Sy [204] reported challenges when Agile teams require input from other non Agile teams (e.g., lawyers, marketing sign offs). Budwig et al. [35] reported that the UX team received unclear requirements and road maps from the business units and as a result the UX team spent excessive time in requirements gathering.Lindstrom and Malmsten [184] reported faster and easier decision making and information access as a result of lack of external dependencies on hardware or people outside the project group. In addition, UX practitioners perceive close collaboration with business people to result in better designs [165]. Brown et al. [33] stated that at the starting point of projects, collaborations between the Agile team and the customer or business team are extremely important to build understand-

ing, develop ideas regarding the software from the user, technological and business perspective, and bring the software into existence.

Practices and Success Factors

The coordination with business units can be achieved via the UX practitioners' involvement with management [5], a facilitator role to handle any work issues with teams outside the Agile development team [204], UX product owner role to act as a central contact point for all cross functional areas in regards to any questions or concerns related to UX deliverable including the business unit and the development teams [35], UX practitioners sharing results with business units [32], and using the planning process to clarify any misunderstandings between business and technical team members [129].

3.4 Challenges and Limitations

This section sheds light on a number of challenges and limitations that were faced whilst conducting the SLR.

1. Since this research is part of a PhD thesis, thus one of the limitation is that a single researcher (student) selected the primary studies and checked it against the inclusion and exclusion criteria. Only a single researcher performed data extraction. Nevertheless, PhD supervisors checked the internal consistency of the protocol to confirm that: the research questions derive the search strings, the data extraction forms allow answering the research question(s), and the data analysis procedure is adequate to address the research questions. In addition, data extraction consistency was checked via two methods: first, data extraction was carried out by myself via randomly selecting a sample of primary studies and subjecting them to data extraction by PhD supervisors. The results were cross checked and any disagreements were discussed and resolved in meetings. Second, a test-retest process were conducted where selected primary studies were randomly selected and a second extraction were performed to check data extraction consistency.

2. One of the main limitations of literature reviews is bias in the selection of publications. To ensure that the process of selection was unbiased, a research protocol was developed in advance that defined the research questions. These questions were used to identify key

words and search terms for identifying the relevant literature. To avoid selection bias, the review process was piloted, especially the citation management procedure and search strategy so as to pinpoint weaknesses and improve the selection process.

3. The extraction process suffered as a result of the method of reporting some of the primary studies. Some papers lacked sufficient information that prohibited their satisfactory documentation in the extraction form and their quality judgement. More specifically, research methods were frequently not adequately described, validity and bias were not always addressed, data collection and analysis methods were often not explained well. Thus there is a possibility that the extraction process may suffer from some inaccuracies. AUCDI studies need to increase the rigour by which they design, conduct, analyse and report their work. Some studies claimed success for their proposed approach without even discussing the case studies, for example, [114, 228].

4. Publication bias is defined as a phenomena where more positive results are published than negative results [162]. This can result in overestimating the effect size in SLRs and under reporting risks [162]. In order to avoid this sort of bias it was decided to classify papers according to success or failure in order to avoid biasing conclusions based on positive results. Something that is worth noting that there is only one paper that reported [209] failure (as well as another case study that reported success) in AUCDI literature whereas 46 other papers reported success. Thus this SLR was based on the available literature. Both academic researchers and industrial practitioners should be encouraged to report failure as well as success.

5. The value of SLRs is highly dependable on the quality of primary studies. However, there is an absence of a consensus on the definition of quality, thus quality assessment of primary studies is very complicated [64]. An additional burden to this complication was the diversity of studies that were covered by the SLR and the attempt to reach a common quality criteria that suits the study type diversity. Sjoberg et al. [264] points out the lack of standard methods to assess the quality of data from qualitative, or mixed qualitative and quantitative research.

   The SLR findings reveal that AUCDI studies rarely provide enough context description to judge the paper's quality. This could be attributed to two reasons: the first reason is that the majority of AUCDI

research are experience reports thus is dependent on anecdotal/ narrative or story telling format of reporting. Nevertheless, context specific information is of great significance to judge the paper quality and understand whether the proposed integration approach can be generalized outside their specific context. The second reason is that out of 71 included papers only 10% were published in journals whereas the 79% that represent the majority of the papers were published in conferences that have usually a paper length limitations that poses a restriction on the amount of information that can be reported. Thus this could lead the reviewer to end up assessing the reporting quality instead of the research quality [64].

6. Since every search engine and digital library differ in the way that search strings are written and how the search is performed. This made searching very exhaustive and time consuming.
7. The proceedings of ESEM were found on the ACM Digital Library for even years and in the IEEE Digital Library for odd years. This also contributed to time consumption and confusion since at first it was assumed that it was biennial, however, after double checking it was discovered that both ACM and IEEE digital libraries needed to be checked for even and odd years.
8. Not all authors were responsive in regards to the emails sent to clarify a missing or unclear details related to their papers.

3.5 Conclusions

In this section, various SLR findings are reviewed and conclusions are drawn.

A state of art SLR was conducted that aimed to identify and classify various challenging factors that restrict AUCDI and explore the proposed practices and success factors to deal with these challenges. The SLR included a total of 71 papers and excluded 80 papers that were published from the year 2000 till 2012. The findings were presented in three stages: first, the SLR findings were quantitatively classified according to year of publication and publication channel. Second, the SLR findings were qualitatively classified into descriptive and content information. Third, AUCDI challenges were explored and their respective proposed practices and success

factors were synthesized and a description and taxonomy of AUCDI challenges and its respective success factors and practices were reported. The SLR drew conclusions in regards to the current state of practice for AUCDI and the future research directions.

Our SLR yielded the following key conclusions:

Conclusion 1.

This SLR is of relevance to industrial practitioners who can utilise the description and taxonomy of AUCDI challenges and corresponding practices and success factors in identifying potential challenges of AUCDI and practices or success factors to deal with these anticipated challenges.

Conclusion 2.

This SLR has a relevance to both academia and industry in regards to future research directions. It reveals the AUCDI research areas that exhibit apparent scarcity and thus need to be further exploited. Examples of AUCDI research areas that need further research are: first, tool support for AUCDI since only three papers Hosseini-Khayat et al. [116], Humayoun et al. [120], Moreno and Yagie [206] tackled this issue. Second, AUCDI in education that was tackled only in Holzinger et al. [114] who presented the concept of extreme usability by integrating usability engineering and XP and taught it in software engineering course. The SLR also revealed AUCDI challenges that are insufficiently researched, for example, difficulty of modularization/chunking and coordination of UX team with business units.

Conclusion 3.

This SLR gave an indication that the current state of research on AUCDI does not pay sufficient attention to controlled experiments or action research. In regards to controlled experiments only Jokela and Abrahamsson [153] paper out of 71 included paper focused on that type of studies. There is also an absence of action research studies that is defined as iteratively involving practitioners and researchers on a set of activities, for example, action intervention, problem diagnosis, and reflective learning [11]. In order to enhance the value and impact of AUCDI research on industry more attention should be given to action research via collaboration of AUCDI practitioners and researchers to set a common research agenda.

Conclusion 4.

This SLR also gave an indication of publication bias in the AUCDI domain since only Najafi and Toyoshiba [209] paper reported failure in AUCDI literature. Both academic researchers and industrial practitioners should be encouraged to report failure as well as success in order to avoid publication bias that can lead to under reporting of challenges.

Conclusion 5.

AUCDI studies need to increase the rigour by which they design, conduct, analyse and report their work. The quality of abstracts, the choice of paper title and keywords needs to be improved in order to enhance the ease of locating their papers. Moreover, papers need to describe research methods more thoroughly and discuss the issues of bias, validity, data collection and data analysis methods.

To enhance the findings of this review, an empirical study will be conducted that investigates current industrial practices for integrating Agile development processes and UCD in order to verify and complement the findings of the SLR. This empirical study aims to identify the common difficulties and concerns that hinder AUCDI attempts and the proposed integration methods. In addition it will compare the findings of this empirical studies with the results of the reported SLR.

3.6 Chapter Summary

This chapter reported on a SLR of the AUCDI literature. Findings were presented in three stages: quantitative data presentation, qualitative data presentation and in the third stage, the data extracted from the primary studies included in this review were analysed and interpreted in order to find answers to the research questions regarding AUCDI challenges, practices and success factors. The analysis and interpretation of the data were reported and conclusions were drawn in regards to the current state of practice for AUCDI. The results of this SLR provide information that can be useful for AUCDI industrial practitioners and academic researchers in order to understand the various challenging factors that may impact AUCDI and the practices and success factors that are being used to deal with the identified challenges. The next chapter will discuss an empirical study that investigated industrial AUCDI attempts.

Chapter 4

Interview Study: A Practitioner Perspective on Integrating Agile and User Centered Design

This chapter focuses on achieving the second objective of the thesis that was discussed in chapter 1, section 1.2. The chapter conducts an empirical study to investigate industrial AUCDI attempts in order to verify and/or complement the SLR findings. The chapter provides an overview of the research method and findings of this empirical study.

4.1 Introduction

This chapter reports on the results of an empirical study that investigated current industrial practices for integrating Agile development processes and UCD in order to verify and complement systematic literature review findings.

4.1.1 Aims

The interview study had the following aims:

- Identify the difficulties and concerns that hinder AUCDI attempts.
- Identify the proposed integration methods.

4.1.2 Interview Questions Overview

The interview is divided into seven categories: company and interviewee, project, requirements, Agile process, Agile and UCD issues, testing and general questions to wrap up. The aim of each category is highlighted as follows:

- **Category 1: Company and Interviewee**

  To gather background information on companies and interviewees.

- **Category 2: Project**

  To gather background information on projects, such as the presence of a UCD practitioner and the Agile method used.

- **Category 3: Requirements**

  To investigate the efforts exerted for requirement elicitation in Agile projects in order to determine the following:

  - The degree of focus on user involvement.
  - The importance given by different companies and projects to usability and/or user experience issues.
  - The variance of the importance of usability and UX issues among different projects and companies and causes of this variance.

  - Techniques used for gathering user interface requirements.

- **Category 4: Agile Process**

  To identify the need for extending Agile practices in order to become a better fit for Agile and UCD integration. In addition, the questions of this category gather feedback on the benefits and shortcomings of software tools in relation to integrating Agile and UCD.

- **Category 5: Agile and User Centered Design Integration**

  To identify the team member responsible for UCD ownership, whether Agile team roles were extended to become a better fit for AUCDI and the effect that this may have on the software's usability or UX.

  This category also aims to identify the method and timing of conducting user interface implementation and how this impacts Agile iterations or release plans or product backlog. Finally, this category investigates the benefits and shortcomings of the different communication methods and tools used for conveying information between the development team and UCD practitioners or the team member playing their role.

- **Category 6: Testing**

  To acquire information about the timing, techniques and participants used for usability testing. This category aims to investigate the method, timing and obstacles for implementing user feedback.

- **Category 7: General Questions to Wrap Up**

  To reflect on interviewee's experience with integrating Agile and UCD, identify advantages and disadvantages of the integration, and identify successful and unsuccessful integration practices.

4.2 Research Method

This section provides the details of the research method used in conducting the interview study. The section discusses the data collection and data analysis methods used.

4.2.1 Data Collection

Qualitative research methods focus on studying complex human aspects such as communication and motivation [163]. Software engineering as a discipline studies a combination of human and technical aspects. Software engineering research utilises both quantitative and qualitative methods so as to benefit from the strengths of both [163]. Qualitative methods provide more informative and richer results through delving into the problem complexity instead of abstracting it. Thus qualitative methods can answer questions that involve difficult to quantify variables particularly those related to human characteristics such as perception, motivation and experience. Qualitative methods can also be used to answer the "why" questions that are addressed via quantitative research. Qualitative analysis suffers from some disadvantages, for example, being exhausting and more labour intensive than quantitative analysis. Moreover, qualitative data is harder to summarize and qualitative results often are "fuzzier" than quantitative ones [163].

Qualitative methods include data collection and data analysis methods [163]. Interviews are usually utilised in collecting qualitative data. Interviews have various objectives, for example, gathering impressions or opinions and collecting historical data participants memories [163].

Semi-structured interviews include a mixture of open ended and specific questions, designed to elicit not only the information foreseen, but also unexpected types of information [163].

The objective of this research was to generate *themes* built on the experience and perspective of industrial practitioners, whether UCD practitioners or developers. Thus a qualitative approach was chosen as the appropriate method for data collection. In this study semi-structured interviews were utilised for data collection since they are flexible, adaptable, allow for following up interesting answers, and provide rich and highly illuminating material [247]. Appendix E shows the list of interview question. An interview guide was prepared and used since it helps the interviewer

to plan the interview. The interview guide is composed of a questions list with notes regarding steering direction for the interview according to changing circumstances [263] as cited in [163].

A number of interview questions were set in advance. The first version of the interview questions were used in conducting a pilot interview. This initial interview highlighted some drawbacks in the interview questions and the interview guide. Most of the drawbacks were related to the length of the interview or the wording of the questions. As a result the number of interview questions were decreased and edited for more clarity. Interview questions are included in Appendix E.

Networking was utilized to reach the initial sampling that involved Agile and UCD practitioners working on projects that utilise Agile methods and developing software that places significant importance on usability and UX. The aim was to explore the space of different environments via interviewing industrial practitioners from a variety of countries, varying size companies, and who develop different types of software at different stages of development. Thus some projects were finished, others were ongoing. Some of the projects involved new versions of software and of existing UI designs and others involved new software and new UI designs. Examples of successful and unsuccessful projects were also included.

Fourteen in-depth, one-to-one interviews (either face-to-face or via Skype) were conducted with 14 participants from 11 companies of varying sizes in five different countries. Those countries are the United Kingdom, Canada, Netherlands, Portugal, and Egypt. Eight interviews were conducted in English and 6 in Arabic, all interviews were translated and transcribed in English. Interviews lasted between 66 and 180 minutes (approximately one to three hours). The interviews covered a broad range of issues ranging from usability and UX goals to incorporating user feedback in Agile teams.

Hand written notes were utilised by the interviewer during the pilot interview. However, after conducting the pilot study it was concluded that recording the interviews and then later transcribing it would make the interviewer more efficient and focused in the interview. Thus all interviews, except the pilot interview, were audio recorded after acquiring the informed consent of the interviewees. This was followed by transcribing all notes in English.

4.2.1.1 Data Analysis-Thematic Analysis

This section discusses the data analysis method. Thematic analysis is "a method for identifying, analysing and reporting patterns (themes) within data" [29, 30]. A theme captures something of importance in the data regarding the posed research question, and represents patterned meaning included in the data set [30]. Thematic analysis minimally organises and describes the data set in rich detail and interprets different aspects of the research topic [29].

Phases of Conducting Thematic Analysis

Boyatzis [29] phases of conducting thematic analysis were adopted as follows:

1. **Familiarisation With Data**

   This phase involved transcribing and recurrent reading of the data, noting down initial ideas. It resulted in familiarity with the breadth and depth of the content.

2. **Generating Initial Codes**

   Codes refer to "the most basic segment, or element, of the raw data or information that can be assessed in a meaningful way regarding the phenomenon" [29]. This phase involved coding interesting data features systematically across the data set and collating relevant data to each code. This phase resulted in the generation of 237 initial codes that identify interesting data features that are meaningful to the studied phenomena. After removing redundant codes and merging codes with similar meaning, 179 codes were reached. All codes were saved in an excel sheet. Examples of those codes are requirements elicitation techniques, usability goals, user experience goals, UCD funds, parallel tracks, upfront requirements gathering, on going requirements gathering, low fidelity prototypes, and high fidelity prototypes.

3. **Searching For Themes**

   This phase involved collating codes into potential themes, gathering relevant data to each potential theme and collating relevant coded extracts of data within the identified themes. Examples of themes that were found are: AUCDI challenges, usability testing methods, and AUCDI practices. An access database was created for the results

of this phase in which a number of fields were kept. Those fields were as follows:

- **A Short Extract:** that contains discrete fragments from the interview transcripts.
- **Category:** that contains themes associated with the short extract.
- **Participant:** that contains details about the participant who mentioned the fragment.
- **Company:** that contains details about the company of the participant who mentioned the fragment.
- **Comment:** that contains notes and represents a form of memoing. Memoing is focused on recording any ideas that occurred during the analysis about possible relationships, reflections and tentative hypotheses that occurred during analysis [43].

Figure 4.1 shows the initial thematic map. It involves 9 themes: perception of development process as AUCDI, responsibility for UX and usability, communication methods between UCD practitioners and developers, knowledge gap between AUCDI literature and industrial practices, quality assurance role in AUCDI, usability testing, AUCDI practices, AUCDI challenges, and implementation of user feedback. Some of those themes have sub themes; for example, AUCDI Challenges has 6 relevant sub themes: management support, upfront design, communication between developers and UCD practitioners, usability testing, UCD practitioner, communication between developers and customers.

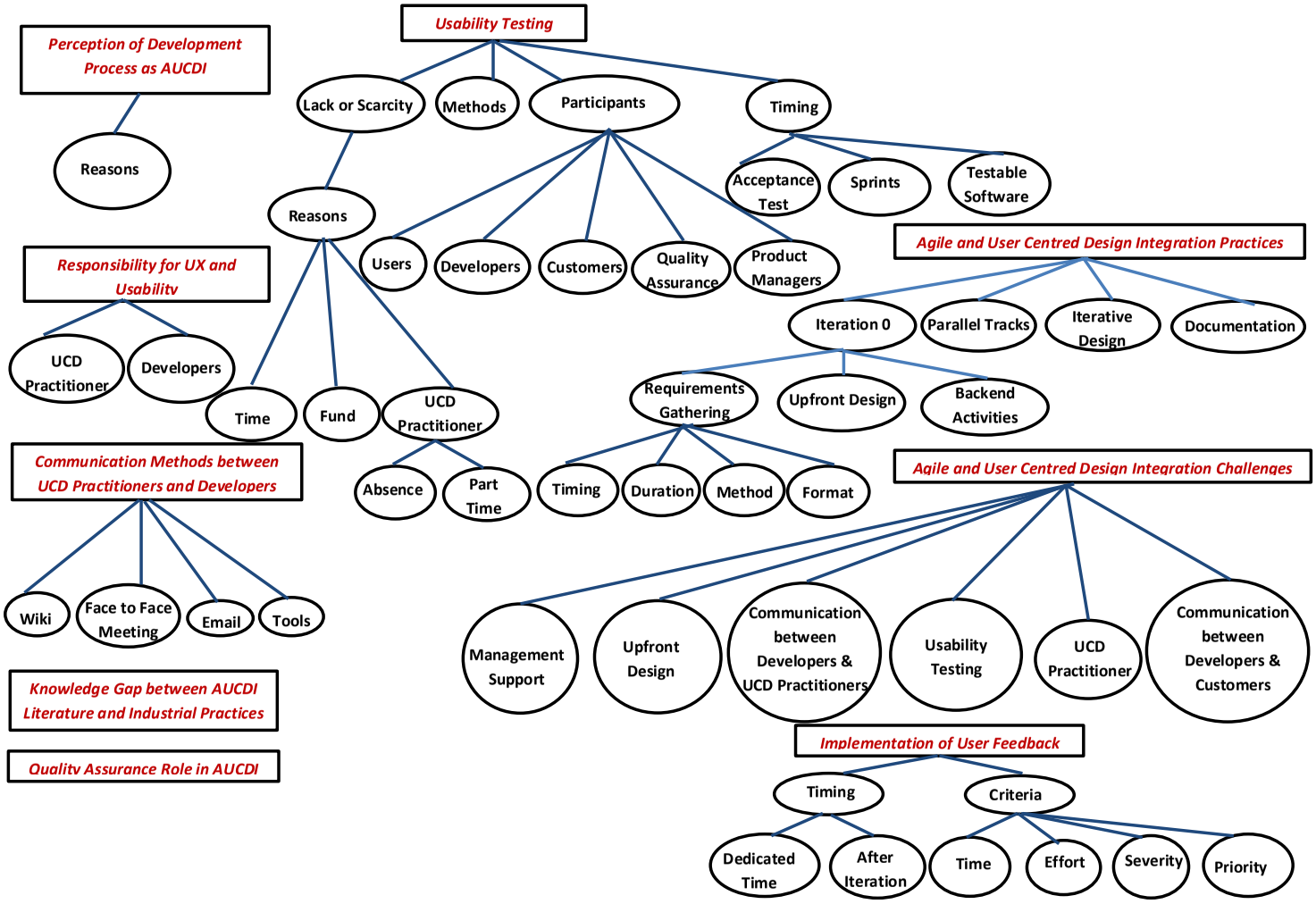

Figure 4.1: Initial Thematic Map

4. **Reviewing Themes**

   This phase involved the refinement of candidate themes to ensure that data within themes cohere meaningfully together and that identifiable and clear distinctions exists between themes. This occurred via checking whether the theme is relevant to the coded extracts and the whole set of data and ended by the generation of a thematic map of the analysis. Thematic maps provide a visual representation of relevant themes. In this phase some themes were excluded due to a variety of reasons, for example, lack of enough supportive data while other themes were merged into each other. Other themes were broken down into separate themes.

   This phase ended by understanding how the different themes fit together and the collective story that the themes convey about the data.

5. **Defining and Naming Themes**

   This phase involved refining each theme specifics and the overall story conveyed by the analysis in regards to the research questions. This phase resulted in generating clear names and definitions for each theme that reflect the essence of each theme and overall themes.

Figure 4.2 shows the final thematic map. It involves five themes: Agile and user centred design integration practices, perception of development process as AUCDI, Agile and user centred design integration challenges, and usability testing.

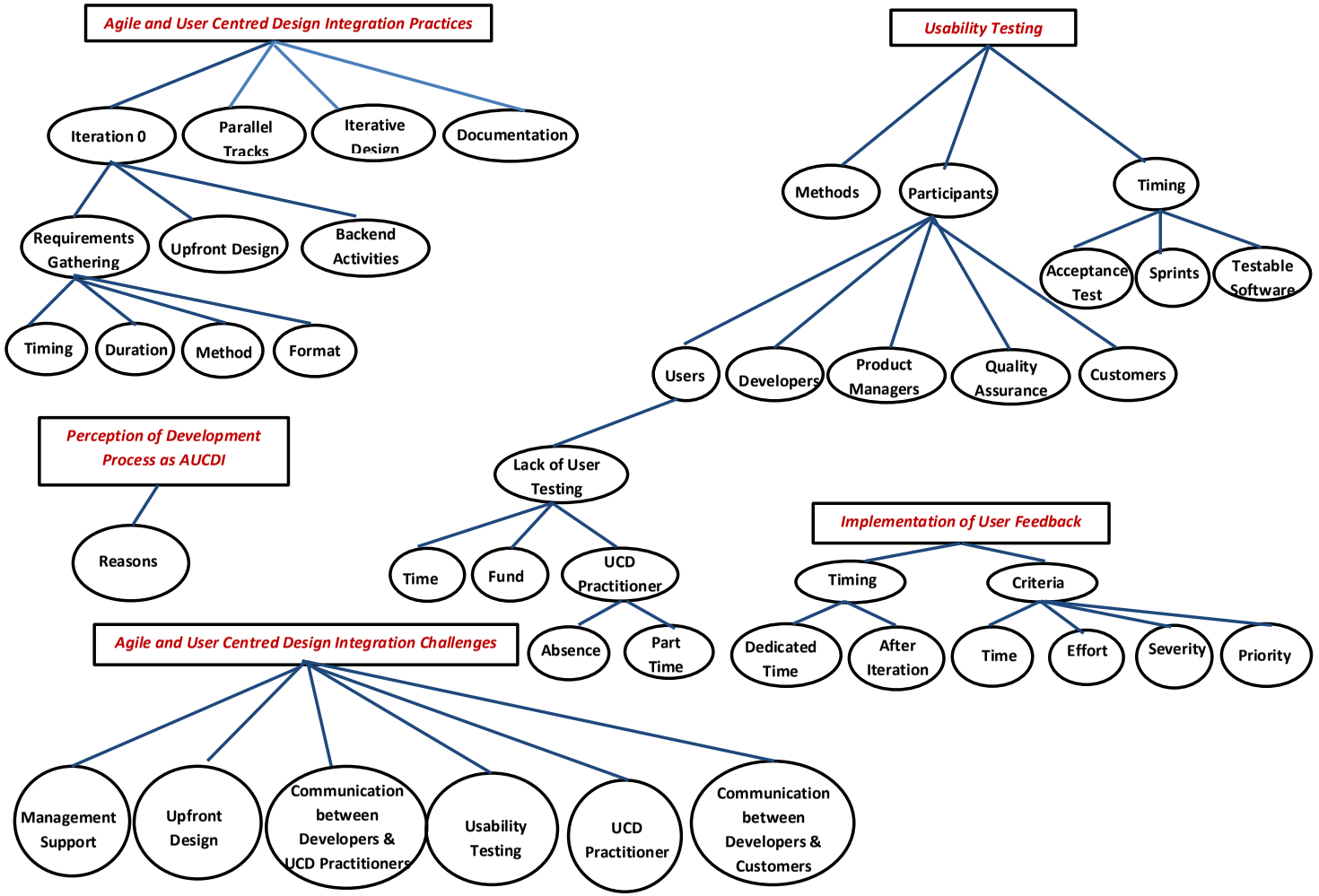

Figure 4.2: Final Thematic Map

4.2.2 Participants' Profiles

This section will introduce participants' profiles and their selection process. For the pilot study one participant was purposefully selected for a number of reasons: his willingness to participate in the study, experience of working as a senior developer in Agile teams, experience in developing a new version of a software whose previous version failed to achieve user satisfaction. This project was the focus of the pilot study.

Since the research was focused on exploring the space of different environment, thus through networking and recommendations, 14 participants were purposefully recruited. The study aimed to interview industrial practitioners from a variety of countries, varying size companies, and who develop different types of software at different stages of development. Moreover, participants were included from projects that developed new software and new user interface designs, and other participants from projects that developed new versions of software and existing UI designs. Participants who worked in successful and unsuccessful projects were also included.

Table 4.1 provides an overview on participants profiles.

Participants were resident in five different countries. Those countries were the United Kingdom, Canada, Netherlands, Portugal, and Egypt. Participants will be referred to as PT1..PT14. The interviewees worked on different development teams for different companies with the exception of PT6 and PT11 who worked for the same company and in the same project but played different roles, and PT2 and PT12 who worked for the same company and in the same project but played different roles in the development team. Participants PT1, PT6, and PT11 all worked for the same company. Full details on participants profiles can be found in Appendix F.

No.	Job	Work Experience Years	Agile Experience Years	Education
PT1	Technical team leader	7	1	Bachelor in CS
PT2	Team leader and a system analyst	3.5	1.5	Bachelor in CS
PT3	Technical architect	10	2	Bachelor in IS and CS
PT4	Principal technical consultant	14	7	Bachelor in IS and CS and masters and PhD in IS
PT5	Independent technical consultant	10	2	Bachelor in CS
PT6	Lead software engineer	8	2	Bachelor in structural engineering and masters in CS
PT7	Senior developer	5	4	Bachelor in CS
PT8	Interaction design group leader	2	1.5	Postgraduate studies in user system interaction
PT9	Software engineer	1.5	1.5	–
PT10	Product manager	17	13	Bachelor in CS and masters in BA
PT11	Product manager	12	6	Bachelor in CS and masters in BA
PT12	Vice president of technology and Scrum master	5	1	Bachelor in CS
PT13	Business analyst	15	8	Bachelor degree and PhD in computing and maths
PT14	Independent technical consultant	12	6	Bachelor in CS and management

Table 4.1: Participants Overview

4.2.3 Project Profiles

This section will introduce the profile of the projects covered in the interviews. The study tried to cover a wider variety of projects; some projects involved new software and new UI designs, while other projects were newer versions of software and existing UI designs. Some projects are finished and others are still ongoing. Projects will be referred to as P1..P12.

Table 4.2 provides an overview on project profiles.

No.	Agile Method	Domain	Duration
P1	Scrum	School educational portal	4 Months
P2	Scrum	Business intelligence application for marketers	4 Years
P3	Scrum	Enterprise resource planning system	1 Year, then aborted
P4	Microsoft Solutions Framework Agile	Governmental data warehouse portal and dash board	10 Months
P5	Mixture of Scrum and Kanban Lean Method (Scrumban)	Project management tool for Agile projects	6 Months
P6	Mixture of Scrum and XP	Educational authoring tool	4-5 Months
P7	Mixture of Scrum and XP	Content management	9 Months
P8	Scrum	Portable global positioning	4-5 Months
P9	Mixture of Scrum and XP	TV guide	1.5 Year
P10	Mixture of Scrum and XP	Internet service provider software	Ongoing
P11	Scrum	Extranet portal	2.5 Years
P12	Microsoft Solutions Framework Agile	Enterprise resource planning system	1 Year

Table 4.2: Projects Overview

Table 4.3 provides mapping details in regards to the different projects and the participants assigned to them.

Table 4.3 shows that participant PT2 and PT12 work in the same projects (P2) in different roles, participant PT2 works as team leader and a system analyst and participant PT12 works as vice president of technology and also acts as Scrum master. Moreover, participant PT6 and PT11 work in the same project (P6) in different roles, participant PT6 works as lead software engineer and participant PT11 works as a product manager. Full details on project profiles can be found in Appendix G.

Participant No.	Project No.
PT1	P1
PT2	P2
PT3	P3
PT4	P4
PT5	P5
PT6	P6
PT7	P7
PT8	P8
PT9	P9
PT10	P11
PT11	P6
PT12	P2
PT13	P10
PT14	P12

Table 4.3: Mapping of Participants and Projects

4.3 Interview Results

This section discusses the interview findings via discussing the main themes that emerged and linking each theme with the relevant interview data in order to illustrate and clarify the results. Some conventions were utilised in reporting quotes; in case one of the quotes include words such as they, he, etc. the meaning was explained between square brackets [ ]. In case the quote includes a sentence and then few irrelevant sentences and then another relevant sentence, the convention sentence one [...]sentence two was used, where [...] signifies the irrelevant sentence.

4.3.1 Theme one: Challenges of Integrating Agile and User Centered Design

The first theme that emerged is related to the reported challenges that were faced by participants in their industrial attempt to integrate Agile development processes and UCD. A number of challenges were reported related to UCD infrastructure, AUCDI process and people involved in the integration process.

4.3.1.1 AUCDI Challenge One: Lack of Management Support to UCD Activities

Participants reported the lack of management support to UCD efforts. This was attributed to a variety of reasons: lack of management awareness of UCD impact on the overall quality of the product, lack of awareness on the importance of UCD practitioner role, tight schedules, and lack of funds. This lack of management support resulted in management reluctance to allocate time, priority or funds for hiring a usability engineer and conducting UCD related activities, for example, usability studies, usability testing, etc.

Participant PT3 discussed management reluctance to allocate a usability engineer for the project as a result of management's lack of awareness of the importance of UCD and UCD practitioner role. Participant PT3 stated that

> I tried to get a usability engineer on board and [...] there was a new project manager and she thought there was no use for a usability engineer and that it was not useful [...] or important for the project [...] she thought it was not worth it and that we are wasting time, effort, and money.

Participant PT10 noted his failure to convince management to allocate funds for conducting usability studies. Participant PT10 also stated that

> At one stage I tried to have a discussion with the company on having some usability studies done on the product and they were not willing to fund.

Participant PT14 did not conduct usability testing on users but rather on customers due to budget constraints forced by management. He stated that

> What prevented me [from testing on users] was not having full time or more than one usability engineer in order to do more research and some limitation with the budget.

Not all teams lacked management support since participants PT11 and PT6 worked in the same project that had full management support to usability. Participant PT11, who worked as a product manager, supported UCD activities via sharing results of competitor analysis, product vision, user requirements, customer feedback and usability testing and involving the development team in analysis of competitors products and prioritiz-

ation of features. Moreover, participant PT11 attended biweekly product demos that allowed her to provide frequent feedback to the development team.

Participant PT11 stated that

> I attended the daily meeting and before we start I told them [the development team] why are we doing this product and what is wrong with the other products [competitors' products] and because they [the development team] investigated the existing products they felt the problem that the teachers are now facing, the feedback from the different education consultants, the feedback from the customers, the feedback from the surveys [...] I let them [the development team] attend [customer meetings] as well [...] I was keen to attend a demo every 2 weeks on the finished product always comment on: this is not easy, this is difficult [...] so they [the development team] realized that it [...] [the product] should be easy to use and that the usability of the product is a key to its success.

Participant PT11 efforts helped the development team to understand users' needs and goals, created shared product and design vision that enhanced decision making in regards to design and usability issues and created a shared UI ownership. Participant PT6 stated that

> The ownership of quality and usability was shared and it was a spirit, the team had as a clear objective, because at the beginning of the project when we were making release planning we put team value and we put ourselves in a state on what is the objective of what we are doing, so [...] [product manager] had a great concentration on usability so this was something totally on our [team] mind.

It can be observed from participants PT3, PT10, PT14, PT11, and PT6 quotes that management support to UCD activities is essential and can either enhance or hinder UCD activities' inclusion in team based planning and scheduling, and execution.

4.3.1.2 AUCDI Challenge Two: Lack of Allocated Time for Upfront Activities

Participants reported the lack of allocated time for conducting upfront activities due to the focus on frequent delivery of software within shorter timescale. As a result some participants struggled to allocate time for upfront activities, for example, analyzing competitive products (PT11), conducting user research and gathering requirements (PT1, PT2, PT3, PT4, PT5, PT6, PT8, PT13), and design activities.

Participant PT8, who worked as an interaction design group leader within the UX team, expressed the difficulty faced by the UX team to conduct upfront activities due to the tight Agile time scales and lack of management support to UCD activities by stating that

> We are pushing as priority, UX takes less priority compared to functionality and this is something that we struggle with right now and also because in this particular project that I am referring to we had a head start [iteration 0] [...] so we are challenged to have this head start before sprint 1 and this is something that is hard to convince the Scrum master and the project team that this is needed.

The challenge of lack of allocated time for upfront activities was extensively discussed in literature, as illustrated in chapter 3, section 3.3, subsection 3.3.1. Although some of the proposed strategies for tackling lack of allocated time for upfront activities were utilised in the studied industrial cases, however, some of the strategies that were proposed in literature were not utilised, for example, little design up-front [3], the design studio [277], and project chartering sessions [283].

4.3.1.3 AUCDI Challenge Three: Communication between the Development Team and UCD Practitioners

Another reported challenge was the communication problems between the development team and UCD practitioners. This had multiple origins as follows:

- Lack of awareness of development team members of the role of UCD practitioner and its importance.
- The absence of a clear road map for communicating between UCD

practitioners and the development team to clarify how UCD activities will fit in the Agile iterative development life cycle.

- Presence of UCD practitioner as a part time rather than a full time team member.

Communication problems were maximized in distributed teams where UCD practitioners and developers were not collocated.

The Agile development process changed the relationship between UCD practitioner (or the team member playing their role) and the rest of the development team [81]. The incremental and iterative nature of Agile development processes requires continuous communication between both parties [59, 193]. This continuous communication involves UCD practitioners sharing design vision, rationale and designs with developers at regular intervals, clarifying design related questions and accepting developers feedback on design issues. The interviews conducted revealed that the absence of continuous communication resulted in frustration among developers, lack of synchronization, delays and bottle necks in the Agile development process.

Participant PT9 was a developer who worked in a distributed team where developers and UX team were located in different countries. Participant PT9 shows an example that reflected the frustration raised among the development team as a result of failure of UX team to share with the development team user needs and goals collected from requirements elicitation phase. When participant PT9 was asked about the technique used by the UX team for user requirement elicitation he replied that

> No [PT9 does not know] and I kind of wish I did but I do not know [...] I wish I knew more about it because our team is entirely frustrated [...] I am totally blind on that.

This resulted in lack of developers' understanding or approval of the design vision as noted by participant PT9 who stated that

> We are trying to understand some of the decisions that were made or have been made because we cannot imagine how this is a good UX but we are not involved in that and at best we can provide some feedback but the feedback we provide is that it [the user interface] does not do what is required but we do not get to affect the requirements very much if at all.

Moreover, the lack of frequent communication of design led to lack of synchronization, delays, and bottle necks in the Agile development process.

When participant PT9 was asked about the UX team's frequency in providing the designs to the development team he noted that

> We wish that they [UX team] would provide it [the design] sooner, the problem is and this is one of the biggest issues, keeping up to date, we had a guide [TV guide design] that was not the most current one and the problem is that the guide did not come in in regular basis [...] and we would check our functionality and say OK works fine here but in fact the most current guide was not the one that we saw, so we were always behind checking the correct user interface experience and there was always a problem so we actually developed our own guide in order to make sure things were working.

This shows how the development team had to exert continuous and extra effort to conduct their tasks as well as recreate UX team tasks in order to be able to complete their work which resulted in delays to the development process. This suggests that teams that operate in parallel need frequent communication to synchronize the work of UCD practitioners and developers.

Although participant PT9 faced difficulties that resulted from lack of shared product and design vision, participant PT11, who worked as a product manager, reported a different perspective that shows the importance of involvement of developers in the design vision process. Participant PT11 was a product manager in the same project as participant PT6 and tried to raise awareness among the development team to the importance of usability issues. This was achieved via sharing the product vision, team vision, results of user studies, usability tests and product feedback. Moreover, the development team was encouraged to attend meetings with users and jointly deciding on high level product goals with the development team.

Participant PT11 efforts resulted in creating a shared team ownership of usability and a shared vision that focuses on users and usability related issues as is evident in participant PT6 quote. Participant PT6, who worked as lead software engineer in the same project as participant PT11, stated that

> The ownership of quality and usability was shared, and it was a spirit the team had as a clear objective, because at the beginning of the project when we were making release planning, we put team value and we put ourselves in a state on what is the objective of what we are doing, so [...] [PT11] had a great

> concentration on usability so this was something totally on our [team] mind.

Another form of communication problems were revealed in power struggles between developers and UCD practitioners [or the team member playing their role] and was reported by participant PT1, PT9, and PT13. Participant PT1 discussed how the part time usability engineer faces power struggles with developers or customers. When asked about what could be improved in his development process he stated that

> Having another usability engineer for consultancy because one usability engineer can sometimes get challenged from developers or customers.

Participant PT13, who worked as a business analyst but acted as the usability engineer, also reported issues related to power struggles between developers and UCD practitioners, she stated that

> The guy who worked in the corporate for I do not know five years prior to that was really upset because he kind of went from being the go to guy to not being the go to guy and the business people went from complaining all the time to praising all the time and that did not work out too well [...] he felt that he did not need anyone that he knew best and he got really upset that he could not ignore me and go ask the user what they want and ignore the exploring stuff.

Participant PT9 also discussed how the UX team ignored the development team's feedback by stating that

> Most of the feedback provided by the development team on UX was ultimately not taken into account, I mean we could say all what we wanted but ultimately no one was actually listening.

The issues related to power struggles reported by participant PT1 and PT13 reflect the importance of acceptance of UCD practitioners by the development team. This acceptance can occur by increasing UCD awareness among team members in order to understand the importance of UCD practitioner's role and what is involved in his work and as a result have realistic expectations from him. Participant PT9 comment indicates the importance of acceptance of development team comments by UCD practitioners since this can lead to shared UI ownership.

The challenge of communication between the development team and UCD practitioners was reported in literature as illustrated in chapter 3, section

3.3, subsection 3.3.4.

A number of strategies were proposed in literature for enhancing the communication between the development team and UCD practitioners were discussed in chapter 3, section 3.3, subsection 3.3.4. However, the studied industrial case studies only utilised a few of these proposed techniques for enhancing the communication between the development team and UCD practitioners. The utilised techniques in industrial case studies included face to face communication aided by scenarios and personas, sharing design artefacts and prototypes, and utilizing information radiators.

4.3.1.4 AUCDI Challenge Four: Conducting Usability Testing

The iterative, rapid Agile time lines, budget constraints, difficulty of accessing users, and lack of full time usability engineers resulted in lack of user testing. This resulted in testing on customers, developers, Quality Assurance (QA) team, conducting hallway testing, or delaying usability testing (sometimes after product is released).

Table 4.4 provides details on the involved parties in usability testing in the different projects. Only 6 projects out of 12 (P2, P3, P6, P10, P11, P12) conducted usability testing on users while the rest of the development teams resolved to testing on development team members, quality assurance team (testers), UCD practitioners, members from other development teams and / or customers.

A number of reasons were provided for not testing on users. Those reasons were the iterative, rapid Agile time lines, difficulty of accessing users, lack of funds, the software being a new version of an existing software or being developed to be used by developers in case of developing project management tools.

Participant PT1 did not test on users due to difficulty of accessing users and lack of funds as stated in the following quote

> Due to difficulty to get teachers and students when is suggested to get them.

When participant PT1 was asked for reasons of testing on customers rather than users, he stated that this was because of

> Time [lack of it] and difficulty of testing on users.

Project No.	Testing on Users	Others Involved in Testing
P1	No	Customers, quality assurance team, product manager, usability engineer
P2	Yes	Users, team leader, customers, testers
P3	Yes	Users, developers, customers, testers
P4	No	Developers, customers
P5	No	Developers, testers, product manager
P6	Yes	Users, quality engineers, developers, product manager, internal project manager, customers
P7	No	Customers
P8	No	Developers, testers, UX practitioners
P9	No	Customers, quality assurance team, developers
P10	Yes	Users, customers
P11	Yes	Users, customers
P12	Yes	Users, customers

Table 4.4: Usability Testing

Participant PT8 did not test on users since his company develops one type of software and the software developed was a new version of an existing software. Participant PT8 conducted peer testing via hallway testing with developers and field test via developers, participant PT8 stated that

> We have a research team that does usability studies and it depends on the goal basically but it ranges from hallway usability testing so that is peer to peer with people on the floor as the most Agile way of testing things [...] we invite people from outside the office to help us in more in depth usability studies and sometimes it is in the office but often we use the car to drive around as in ourselves or we invite people to join us, so that is another really quick way to test our products.

Participant PT5 developed a Project Management (PM) tool for Agile projects, a visual dash board for monitoring and tracking Agile teams' project status. He did not test on users since the software developed was for developers and thus he thought that the development team will be sufficient to test the software. Participant PT5 stated that

> The advantage that we have is that we can relate to the user easily because the users resemble us because they are technical people and we wanted to make an application that we ourselves will like to use, so this was our advantage.

A number of projects (P1, P2, P3, P5, P6, P8, P9) involved the quality assurance team (testers) in testing the software functionality and usability.

Participant PT1 declared that QA team tested both functionality and usability every two days and provided the usability engineer with feedback. Participant PT1 stated that

> QA tested every two days since they tested with every build which is every two days and could give feedback to usability engineer so were checked and validated and corrected, feedback was right away as defects [reported right away].

Participant PT14 conducted usability testing regularly on customers where they were observed while using the low fidelity or high fidelity prototypes. He stated that

> Every two weeks we made a demo to customer this occurred all along until end of four months.

The challenge of conducting usability testing was reported in literature as illustrated in chapter 3, section 3.3.

Although, literature on integrating Agile and UCD that is discussed in chapter 3, section 3.3.5 offers a number of strategies for usability testing, for example, preparing for user research [136], utilizing discount usability engineering techniques [56, 76, 89, 156, 198], and conducting remote usability testing [59]. However, the studied industrial case studies only resolved to testing on development team members, quality assurance team (testers), UCD practitioners, members from other development teams and / or customers rather than users to face the tight Agile time lines, difficulty of accessing users, and lack of funds for conducting usability testing.

4.3.1.5 AUCDI Challenge Five: Communication between the Customer and the Development Team

A number of reported challenges for integrating Agile development processes and UCD were related to customers. These included misrepresentation of customers in Agile teams and lack of customer awareness on Agile methods.

Misrepresentation of Customers in Agile Teams

Participants reported that customers were misrepresented in Agile teams: customers did not present the wider user population and were not aware

of user needs, problems or goals. Some participants remarked that customers did not have the authority to take decisions when needed. Misrepresentation of customers in Agile teams led to erroneous decision making in regards to user needs, problems or goals.

Participant PT3 expressed the lack of customer authority and knowledge in the domain by stating that

> I think what would have helped is to educate the customer,[...] he just had some practices which he learned throughout the years, [...] he did not have in my opinion the best inventory practices [...] he did not have the authority or the knowledge.

Participant PT14 also expressed the lack of customer domain knowledge which led to erroneous decision making in regards to user needs, problems and goals by declaring that

> Sometimes the customer did not have the real knowledge about how things really work [...] Yes that created some problems because in some cases we had to redo some things.

Lack of Customer Awareness on Agile Methods

Lack of customer awareness on Agile methods was shown in customer reluctance to provide continuous and frequent communication with the development team. This led to delayed customer feedback and made it harder for the development team to achieve customer and subsequently user satisfaction. Some participants reported that customers lack of frequent and continuous communication at the requirements gathering phase and usability testing phase led to delayed feedback that hindered Agile team's productivity.

Participant PT1 expressed the importance of customers awareness on Agile methods to achieve continuous involvement and frequent communication

> Agile is not suitable for projects in which customer does not understand what is Agile, customers should be educated and know that they should be more involved and give frequent feedback.

Participant PT9 expressed the effect of customer latency in providing feedback to the development team by declaring that

> We would ship something to partners [customers] every month and they would not start using it until another month so it will be two months [...] it was frustrating to us as developers but

> we could not convince them [customers] that if this was just a shorter cycle it would be much better.

Participant PT2 discussed the failure of customers in abiding by the tight Agile time lines in requirements gathering by stating that

> Requirement gathering is painful for customer because he is a business customer so he has a lot of work to do. So always asks about can you send this to me by mail and I will get back to you.

The challenge of communication between the customer and the development team was reported in literature as illustrated in chapter 3, section 3.3. The proposed techniques for tackling this issue are: trust [261], particular customer features [261], planning ahead [261], common vision [261], communication techniques [32], and customer perception [261].

Planning ahead, trust, customer features, communication techniques, and perception were utilised by the investigated industrial case studies.

4.3.1.6 AUCDI Challenge Six: UCD Practitioners

Another set of challenges that impacted the integration efforts were related to UCD practitioners. These included the absence of UCD practitioners in Agile teams and their presence as a shared resource.

Absence of UCD Practitioner

Some of the participants' teams included a UCD practitioner while others did not. Table 4.5 shows 8 out of 12 projects did not include a usability engineer. As a result this role was played by another team member, for example, developers, business analysts, technical consultants or designers. This raises concerns on whether those team members were qualified to play this role. This resulted in delays in the development process, difficulty of inclusion and prioritization of usability or UX features in planning and scheduling activities, discarding some UCD activities such as user testing, and compromising the usability or UX of the software.

Participant PT12 was discussing the implications of the lack of UCD practitioner and how this role was played by developers and the impact that this had on the project progress, he stated that

> Most of my time is wasted in UI [...] people [developers] who ask for time extension always say due to UI problems because

Project No.	Presence of Usability Engineer (UE)	Team Member Performing UE Role	UE Status
P1	Yes	–	Part Time
P2	No	Developers	–
P3	No	Graphics Designer	–
P4	No	Principal Technical Consultant	–
P5	No	Developers	–
P6	Yes	–	Part Time
P7	No	UI/UX Designer	–
P8	Yes	–	Part Time
P9	Yes	–	Part Time
P10	No	Business Analyst	–
P11	No	Independent Technical Consultant	–
P12	No	Graphics Designer	–

Table 4.5: Role and Responsibility for Usability and User Experience

> they are developers and I have no UX designer so they take time, sometimes due to inconsistencies someone may overwrite on the other [UI work].

Participant PT5 stated that the lack of UCD practitioner led to lack of formal usability testing, he stated that

> The problem with implementation up till now is that we do not have formal usability methods, we need to have a usability engineer who works with some formalization for usability methodologies up till now everything [usability issues] was implicit.

Participant PT3, whose team lacked a UCD practitioner, replied to a question about what could be improved by saying

> We were not thinking of usability we were actually thinking of basic functionality and that was just what we wanted reaching a level of functionality which the user could use.

Participant PT1, discussed the advantages of having a UCD practitioner since it positively impacted the performance of developers due to taking the burden of UCD activities off them. Participant PT1 stated that

> Having a usability engineer and graphics designer [...] advantages were that people [the development team] were concen-

> trated and made research on UCD issues and took this burden off developers [...] we also have better feedback from customers and usability engineer was better equipped to justify decisions.

UCD Practitioner Workload

Table 4.5 shows that only four teams (P1, P6, P8, P9) included a usability engineer. Although Project P7 had a UI/ UX designer and projects (P3, P12) had a graphics designer yet those team players were only responsible for interface design activities and not usability engineering activities. Those four teams that included a usability engineer had him as a shared resource and acted as a part timer. Although this is a rather common situation, arguably an Agile development process adds more burden to UCD practitioners due to the Agile nature that requires team members to attend a number of meetings for all the teams they are involved in. Moreover, usability engineers do not have enough time to finish their work due to the rapid and iterative Agile time lines. The consequences of this included discarding some UCD activities, for example, gathering requirements from users and instead gathering it from customers (PT14). Furthermore, UCD practitioner workload led to slow response from usability engineers that eventually led developers to play the role of UCD practitioner in case of his absence (PT6).

Participant PT14 had a part time usability engineer who was contracted to work 40 hours per month, when asked about feedback on this model of work he stated that it led to lack of requirement gathering on user needs and subsequently failure of the software to satisfy users

> [...] the funds for design were cut out because the customer wanted everything a lot cheaper and my administration thought that that was the person to get rid off and I could not really do anything about it, just 40 hours per month was the best thing I had [...] The disadvantages because we had to drop something like the usability engineer [as a full timer] and like some usability requirement gathering and some other parts we lacked some more knowledge about the user [...] we were replacing an existing one [software] and we failed and we were trying to make something similar to the previous one and at the same time innovative and that was really not very easy to do and not very interesting to do at the end result we should really have a usability requirement gathering in order to address the project because in some cases the customer was saying to us no no

> they really enjoy how this application is built and the end user would tell us no no this sucks.

Participant PT6, lead software engineer, expressed the impact of not having a full time usability engineer by stating that

> The people who work in graphics and usability are shared resources [...] They are actually allocated for a while but we need to arrange for more time to sit with him [...] the problem in this is that he [usability engineer] can leave us before the end of the project.

The challenge of absence of UCD practitioner and their presence as a shared resource was reported in literature as illustrated in chapter 3, section 3.3. A number of methods were proposed in literature for dealing with this challenge, for example, mentoring process [5], office hours approach [76, 181], decreasing UCD practitioner workload [76, 181], and distributing UCD practitioner workload [283]. None of those proposed strategies was utilised in the industrial case studies and the challenge was dealt with by the studied projects via assigning another team member to perform the tasks of UCD practitioner. This team member could be a developer, business analyst, technical consultant or designer.

Thus it can be concluded from theme one that AUCDI challenges have various causes related to three dimensions: UCD infrastructure, AUCDI process, and people involved in the integration process. First, UCD infrastructure is exemplified in challenge one; lack of management support to UCD activities. UCD infrastructure is embodied in the allocation and utilisation of dedicated funds for conducting UCD related activities as well as management support to UCD activities. Second, AUCDI process is exemplified in challenge two; lack of allocated time for upfront activities and challenge four; conducting usability testing. AUCDI process is related to a development life cycle that embraces the planning and inclusion of UCD activities within the iterative and incremental Agile development process. Third, people is related to the communication and attributes of those involved in the AUCDI process, for example, customers, users, developers, and UCD practitioners. It is exemplified in challenge three; communication between the development team and UCD practitioners, challenge five; communication between the customer and the development team, and challenge six; UCD practitioners.

Those three dimensions affect AUCDI endeavors and accordingly any challenges related to them need to be tackled for successful integration of Agile

and UCD.

4.3.2 Theme Two: Agile and User Centered Design Integration Methods

The second aim of this empirical study is to identify the proposed integration methods. This section discusses the emerged themes regarding integration methods for Agile development processes and user centered design. Those include methods for upfront design, requirements gathering and presentation, synchronizing the activities of developers and UCD practitioners, usability testing, iterative design and lightweight documentation. Those methods were listed below according to the order of their utilisation in the Agile development life cycle in order to achieve Agile and UCD integration.

Iteration 0

The different Agile teams faced challenge two; lack of allocated time for upfront activities via allocating an iteration 0 for UCD practitioners or the Agile team member playing their role to gather requirements, understand users, user goals and context of use, prioritize features and conduct upfront design in order to achieve holistic design vision. Iteration 0 was utilised by developers in working on back end features such as selecting system architecture and development environment.

Iteration 0 was used by participants for analyzing competitive products (PT11), requirements gathering (PT1, PT2, PT3, PT4, PT5, PT6, PT8, PT13), and upfront design. The length of iteration 0 varied among the different teams, since some teams, for example, PT6, had more than iteration 0. However, a number of participants (PT1, PT2, PT5, PT9, PT10, PT11, PT13, PT14) noted that although requirement gathering occurred at iteration 0 yet the requirements gathering process was ongoing. Whereas one participant, PT7, stated that requirements gathering occurred later in the development process, this was attributed to the fact that he was developing a subsequent version of a particular software.

Participant PT8 discussed activities performed in iteration 0 by stating that

> The first sprint for development was focused on non UI features basically there was no team work needed for it but later on there was and that was good because the designer had time

> to design and write special cases [...] developers were working on back end [...] mostly about architecture and server side complexity that they need to solve. It [iteration 0] was about optimizing code and preparation mostly for the functionality.

Participant PT6 noted that his team utilised more than iteration 0.

> Actually we had more than one iteration 0, we took a while to collect UI requirements and functional requirements and comparing with other products.

Iteration 0 was the method used by participants to tackle the challenge of lack of allocated time for upfront activities. It was used to gather requirements, prioritize features and conduct upfront design. Iteration 0 was utilised in literature as illustrated in chapter 3, section 3.3.1.

Requirement Gathering Methods

Different methods were used for requirements gathering, for example, interviews (PT2, PT3, PT4, PT6, PT10, PT13, PT14), contextual inquiry (PT14, PT6, PT13), observations (PT5, PT14, PT13), and surveys (PT4, PT6, PT10).

Participant PT1 obtained requirements via the products manager who acted as a proxy customer and developed requirements from lessons learned from previous projects. He justified the lack of users involvement in requirements gathering by the fact that he was developing an off-the-shelf product. Participant PT1 noted that

> We got lessons learned from previous projects and we hired a person as a customer who represent all stakeholders and we gathered requirements from him and he was product manager and responsible for product vision.

Participant PT8 stated that requirements were set via the project scope and brief that was written by the product manager and used by the project manager and the interaction design group leader (participant PT8) to come up with the requirements. He justified the lack of user involvement in requirement gathering by the fact that the company is specialized in navigation software and as a result the development team worked in a lot of similar projects, participant PT8 stated that

> The project brief was written and the user requirements were driven from the business goal and this was a joint effort from the interaction designer and the project manager [...] as you see the business of the company is navigation.

Participant PT11 gathered requirements via surveys, she stated that

> These surveys were sent directly to school teachers. We have a training team [...] that is responsible for giving training on our application to end users and the teachers and through this team they made some surveys to the teachers and we collected their feedback.

Table 4.6 provides details on the involvement of users in requirements gathering in the different projects. Only 6 projects out of 12 gathered requirements from users while the rest of the projects gathered requirements from customers, product managers, product owners, and/or lessons learned from previous projects. However, this does not suggest that the other 6 projects did not gather requirements from users since the customers usually were responsible for that.

Project No.	**Gathering Requirements from Users**	**Source of Requirements**
P1	No	Product manager and lessons learned from previous projects
P2	No	Customer
P3	Yes	Users, customer
P4	Yes	Users, customers
P5	No	Customer
P6	Yes	Users, product manager, customers
P7	No	Product owner, lessons learned from previous software version
P8	No	Project manager(Scrum master), interaction design group leader
P9	N/A	N/A
P10	Yes	Users, customers
P11	Yes	Users, customers, product managers
P12	Yes	Users, customers

Table 4.6: Involvement of Users in Requirements Gathering Process

A number of teams were keen for Agile developers to become involved with users. This occurred via a number of methods, for example, attending contextual inquiry sessions, requirements gathering workshops, etc. This reportedly led to a number of advantages: increased developers' awareness and knowledge of users and usability issues and increased motivation for the development team to focus on UCD and act as user advocates.

Different methods were used for requirements gathering, for examples, in-

terviews (PT2, PT3, PT4, PT6, PT10, PT13, PT14), contextual inquiry (PT14, PT6, PT13), observations (PT5, PT14, PT13), and surveys (PT4, PT6, PT10). These same methods were utilised by a number of practitioners and academic researchers as illustrated in chapter 3, section 3.3.1.

Requirements Presentation

The gathered requirements in iteration 0 were presented in different forms, for example, personas (PT1, PT5, PT13), low fidelity prototypes that took different forms. Forms of low fidelity prototypes were: paper prototypes (PT13), paper sketches (PT14), electronic sketches (PT1), power point (PT2) or high fidelity prototypes (PT2, PT3, PT4, PT5, PT7, PT14), scenarios (PT3, PT9), or narratives (PT3, PT8, PT10).

Participant PT5 expressed the advantages of personas and how they help in bringing the development team closer to the users' mentality. Participant PT5 stated that

> Personas are very useful and had a lot of advantages it gives a human touch for requirements because at the end there is a human being who will work with this stuff. When all team communicate we can feel this human touch and feel at the end how will the user who has a different mentality, knowledge, skills, and IQ level will be able to use this tool.

The gathered requirements in iteration 0 were presented in different lightweight forms, for example, personas (PT1, PT5, PT13), low fidelity prototypes (PT13, PT14, PT1, PT2), high fidelity prototypes (PT2, PT3, PT4, PT5, PT7, PT14), scenarios (PT3, PT9), or narratives (PT3, PT8, PT10). These same methods were utilised by a number of practitioners and academic researchers as illustrated in chapter 3, section 3.3.1.

Parallel Tracks

Parallel tracks [203] were utilised by participants (PT1, PT2, PT3, PT5, PT7, PT8) in order to coordinate the work between UCD practitioners and developers via organizing work in two parallel and interrelated tracks. Parallel tracks organizes work around a number of cycles: cycle n, involves developers working on back end features and UCD practitioners working on design of features that will be implemented by developers in cycle n+1 via building prototypes. These prototypes are used to test the design, conduct design test, and fix prototype errors. Cycle n+1 involves UCD practitioners presenting the designs from cycle 1 to developers in order to implement them and UCD practitioners working on prototyping and

usability testing for cycle n+2 features. These cycles continue until design goals are achieved. Parallel tracks [203] were utilised by participants (PT1, PT2, PT3, PT5, PT7, PT8) in order to coordinate the work between UCD practitioners or the team member playing their role and developers via organizing work in two parallel and interrelated tracks. Parallel tracks were used by a number of industrial practitioners and academic researchers as illustrated in chapter 3, section 3.3.4.4.

Usability Testing

Section 4.3.1.4, table 4.4 provided details on the involved parties in usability testing in the different projects and showed that the tight Agile time lines, budget constraints, difficulty of accessing users, and lack of full time usability engineers led some participants to conduct testing on customers, developers, quality assurance team, or conduct hallway testing. As a result, only 6 projects out of 12(P2, P3, P6, P10, P11, P12) conducted usability testing on users whereas the rest of the development teams resolved to testing on development team members, quality assurance team (testers), UCD practitioners, members from other development teams and / or customers.

Timing of Conducting Usability Testing

Not only did the different teams vary in regards to the participants involved in usability testing but also in regards to the timing of conducting these usability tests.

The timing of conducting these usability tests varied. Some participants (PT2, PT6) adapted acceptance testing in XP projects or demonstration sessions in Scrum projects to include time for usability testing. Other participants scheduled testing to occur at certain time, for example, at the end of iteration or sprint (PT2, PT7) or weekly (PT9).

Participant PT2, who worked in project P2, stated that testing occurs after every iteration

> They [users] use that software after it goes into production and this happens after every iteration.

Participant PT6, who worked in project P6, noted that testing occurred frequently along the project

> In the middle of the project two or 3 times, those users take the tool, we either demo it to them or we gave them access to the location of the builds and we tell them to check it out at any

time and this actually happen.

Participant PT7 conducted usability testing at the end of every sprint. Participant PT7 declared that

> Last week of every sprint was used for testing new sprint.

Participant PT9, who worked in project P9, declared that usability testing occurs weekly by the development team and monthly by system integration. Participant PT9 stated that

> We [developers] did weekly iterations in which we would test with our own team and after four weeks every months we would send to system integration people and after system integration people have it for a month they would send it to our other partners and that when the two months delay happen.

Participant PT9, also stated that testing on users will be conducted later on in the development life cycle. Participant PT9 stated that

> It will be a combination of QA specialists and system integration and test team and after that it will get shipped to a larger system to put it on a system that is almost live and they will roll it to some customer and then try it with them for a while and eventually it gets out to a larger base.

The timing of conducting usability testing varied also in literature as reported in as illustrated in chapter 3, section 3.3.5.2 in Detweiler [59], Ferreira [78], Kane [156]. Some researchers suggested fitting usability testing in other Agile tests, for example, XP acceptance testing sessions [78] or Scrum demonstration sessions [156]. Hussain et al. [129] extended mandatory unit testing in XP via adding usability specific test cases in order to define an application's UX. Carbon et al. [36] suggested four adaptations to XP in order to improve XP focus on usability. These four adaptations were: integrating usability criteria in user stories, introducing usability stories, deriving acceptance tests from the integrated usability criteria and usability stories, and using acceptance testing for conducting usability evaluations. Albisetti [5] introduced a mandatory UI reviews as a gate keeping tool, where two sign offs were set one for code and one for UI. McInerney and Maurer [193] reported that TB team conducted usability testing at least once for every internal release (three months). However, Ferreira et al. [81] perceived iterations and releases completion as an opportunity to frequently test the usability of the software.

Forms of Usability Testing

Forms of usability testing varied between low fidelity (PT10, PT14) or high fidelity prototypes (PT1, PT2, PT3, PT4, PT5, PT7, PT6, PT8, PT9, PT10). The forms of usability testing that were reported in literature were reported in chapter 3, section 3.3.5.1.

Iterating Design

Participants varied in regards to the timing of iterating the design as a result of usability testing.

Some participants (PT1, PT2) allocated time from every iteration for fixing functionality and usability bugs. Other participants iterated design in the same iteration that they got the feedback in (PT2, PT3, PT4, PT5, PT6, PT7, PT8). Some participants postponed modifications to the following iteration or sprint in case the modifications require considerable time or are of low priority or low severity (PT1, PT2, PT6, PT8, PT14). Some participants (PT1, PT8) noted that some modifications were not implemented in case its gain is less than its cost.

Participant PT1 allocated part of every iteration for bug fixes and considered usability errors as part of bugs. Participant PT1 stated that he works on usability errors

> Right away and we have allocated a part of each iteration to bug fixes and this was time allocated [...] sometimes user interface feedback was done right away and sometimes it was moved to next iteration according to comment in what and when and its size all that are controlling variable and some feedback were not actually implemented because its gain is less than its cost and this was a team decision.

However, participant PT1 also noted that

> In case the error requires additional time, it is moved to next iteration in case it is worth the effort.

Participant PT2 determined the timing of fixing both usability and functionality bugs according to the source of the bug and customer opinion. In case the developers were responsible for the bug or the customer needed the bug to be fixed right away then that was performed. Participant PT2 stated that

> We try to leave two days at the end of every iteration for bug fixing and this was not made at first but we later discovered we

> needed to include it [...] According to bug if we [development team] are responsible for those bugs we fix them right away. If it is a change request we estimate its time and may move it to next sprint, however, if she [customer] insist on having it in current sprint we inform her that we have to remove something to include her request so according to bug severity and amount of work required and same scenario if the feedback is given by a user to the customer and conveyed by the customer to us.

Participant PT3 did not have an allocated time of every iteration for implementing user feedback and as a result he implemented user feedback right away. This resulted in postponing the implementation of features and lack of time to finish working on features set per iteration. Participant PT3 stated that

> The user feedback was top priority, implement it as much as we [development team] could and try to push the features back, the problem is the iteration was not enough to implement both the user feedback and the new features.

Participant PT6 iterated design on the same day if the feature is important and time manageable and could be fixed quickly, however, he postponed fixing features that will consume too much time into later iterations. Participant PT6 stated that

> Some issues were performed on the same day, if it is a nice idea and do not contradict with the whole project [...] and some other stuff take a little longer time according to the nature of the feature [...] naturally that could take a long time, we needed to take some time to re-plan it, but the team is self motivated so if something could be finished quickly they just finish it.

Participant PT8 iterated design according to its priority and severity

> It depends also on the topic of course, there is priority list being created and of course also depending on the company priority that decide what is done [...] and depending also on the severity of the bug that determines if it is solved right away or in another sprint or not at all.

Participant PT14 iterated design in the same iteration or the following one according to the priority and importance of the raised issue. Participant PT14 stated that

> Normally we would implement as soon as possible in terms

> of placing that into the next iteration that we have if we have an iteration that was really in the middle we would only incorporate it if we were working on that particular feature or else it would go into the next iteration [...] we were prioritizing every part and saying OK this is an issue which level of issue it is very important OK low importance so at the end we will address that, it is highly important because this is making the usability of the application really worrying and no one wants to use that OK so next interaction.

The challenge of short time available to iterate design with user feedback was handled in literature via dedicating cycles for working on user feedback and incorporating it into the development life cycle [193] and utilizing the UX practitioner to act as an Agile customer to ensure incorporation of user feedback into the development life cycle [204].

Documentation Methods

A number of issues needs to be documented, for example, user requirements, design rationale [193], prior designs, usability testing procedures, designs to be implemented, and their associated delivery schedule and usability testing results [272]. Different methods were used by the different teams for documentation purposes, for example, Wiki (PT3, PT6, PT7, PT10, PT14), Excel (PT3, PT6, PT7, PT10, PT13), and information wall (PT1, PT8, PT9, PT10, PT6, PT13). All these methods are simple, easy, and lightweight.

Wiki

Wiki was used by participants (PT3, PT6, PT7, PT10, PT14) to create a shared product vision between customers and the development team. It provided easy and central access to user requirements, design rationale, prior designs, usability testing procedures, designs to be implemented and their associated delivery schedule, and usability testing results.

Participant PT6 stated that Wiki was used for documenting the following

> Stories and its acceptance tests and tasks that we reach after we put detailed design and any models that we did we either photograph it and put it on Wiki or use visual studio diagramming tools.

Excel

Excel was used by participants (PT3, PT6, PT7, PT10, PT13) for document-

ation and back up purposes of development team efforts. Excel sheets were viewed by managers to monitor and track development team progress. Participants stated that Excel was easy, simple, and sufficient.

Participant PT3 used Excel for documentation purposes and communication with managers. Participant PT3 stated

> The excel sheet was basically a status report that I accumulate to give to my manager.

Participant PT10 used excel in documentation due to its simplicity. In regards to a question on how participant PT10 evaluates excel as a documentation tool, he stated that

> I believe in simple tools to keep the process simple so excel did more than enough for me.

Participant PT13 utilised Excel as a documentation method for backup purposes.

Information Wall

The information wall was used by participants (PT1, PT8, PT9, PT10, PT11, PT13) in temporary documentation. It helped in communication via acting as a central point that brought the development team closer. This occurred via facilitating team discussions and negotiations, providing a shared progress vision, monitoring progress. The information wall helped in increasing team motivation and focus on current tasks, and helping in establishing mutual understanding, consolidated opinions, and team vision.

Participant PT1 noted that the wall provides a shared progress vision, motivates the development team and helps in setting expectations. Participant PT1 stated

> The wall was very good in stories and tasks because it gave vision on progress and motivated people and helped people who are dependent on walls like QA people because they knew when to expect a build.

Participant PT10 declared that the information wall helped in monitoring purposes and increasing team focus on current tasks. Participant PT10 stated that

> It allowed us to focus on what we were doing, we were very clear on what we were doing and what we were not doing so I

> would say it is more than just monitoring it, it created focus for us.

Participant PT11 commented that the information wall is only temporarily useful for monitoring purposes and increasing team focus on current tasks, however, since it gets erased, as a result spread sheets were a more convenient alternative for permanent documentation purposes

> The information wall is not kept across products [...] we did not reuse the data because it was papers on walls and then after the product ended we did not keep the data, we need to have this data copied in a digital format in case we need it later [...] I get into the room during the demo time I have a look at the information wall that tells me the bug status, that tells me what went right and what went wrong in every iteration so it gives me everything I need once I enter the room but for keeping the records we designed time sheets where the Scrum master or the project manager handles copying the data into sheets so we can use it later.

Participant PT13 stated that the information wall acted as a central point that helped in enhancing team communication and negotiation, focus, feature ownership, and providing a shared vision for tracking progress. Participant PT13 elaborated on the advantages of the information wall by stating that

> One of the reasons of how I got people who did not talk to me initially to come and talk to me because they thought it was a mistake [what is included on the wall] they would say no no no this is a mistake this is more like this and that and so we changed it and we put their hand writing on a card board and people would take ownership and that meant when people walked past they knew what was happening [...] everyone was focused and was prioritized and bargained.

Participant PT8 utilised information wall and stated that it helped in visualizing work done and not done and acted as a central communication point for team members

> So having walls with sticky notes [...] it is in a way an alternative to the burn down chart and it is a better way to visualize the amount of work that is done and still needs to be done [...] the sticky notes how do I say this is a better alternative to monitor because it is physical, it forces also project members to go to

> the wall and look at the tasks and either move the tasks when it is done or select new task when they finish one and also have some effect as coffee machine people are attracted to talk to each other, it is also a control point in stand up meetings.

Although the Agile Manifesto [14] values working software over comprehensive documentation and states that "the most efficient and effective method of conveying information to and within a team is face-to-face conversation", however, Documentation is necessary for achieving a successful integration of Agile development processes and UCD. Participants resorted to simple, easy and lightweight documentation methods in order to cope with the tight Agile time lines. Those were Wiki, Excel, and information wall.

4.3.3 Theme Three: Perception of Development Process as AUCDI

A question was posed to participants regarding whether they consider their model as an Agile UCD model or not. Answers varied, some participants considered their model as an Agile UCD model while others did not as it can be perceived from the following comments.

Participant PT1 considered his development model as an AUCDI model due to the presence of a usability engineer and graphics designer although: first, both the usability engineer and graphics designer were not involved in requirements elicitation, participant PT1 himself, stated that

> Graphics designer and usability engineer had no input in requirement gathering although we have user interface requirements.

Second, requirements were set via product manager and lessons learned from previous projects rather than gathered from users as discussed in table 4.6. Third, the development team did not conduct usability testing on users but rather on customers, quality assurance team, product manager and usability engineers as discussed in table 4.4.

Participant PT7 considered his development model as an AUCDI model due to the utilization of personas although: first, he did not gather requirements from users but from product owner and lessons learned from previous software version. Second, the development team did not conduct usability testing on users but rather on customers as shown in table

4.4. Participant PT7 stated that

> I say it was user centered because personas were the driver of our mentality, satisfying the users is the core value there was no core value to have an application that was technically solid or has certain architecture, no we were more concerned that it would satisfy customer need that we can relate to so the user was in the center in this process.

Participant PT10 considered his development model as an AUCDI model although he was not aware of that before the interview, however, the questions posed made him realize the amount of activities he went through in order to achieve user satisfaction. Participant PT10 stated that

> I consider it [his development model] much more UCD having talked to you tonight than I ever did before, I would have not considered it an Agile user centered model, talking to you tonight and talking about details we were much more responsive to users I think more than I realized!

Participant PT12 considered his development model as an AUCDI model because he was responsive to change and frequently delivered software although these two qualities signify any Agile project according to the Agile Manifesto [14]. Participant PT12 stated that

> I never refused change of request, I always make frequent deliveries and we try to apply agility so all my work is customer and user focused.

Although participants (PT1, PT7, PT10, PT12) considered their development model as an AUCDI model, other participants (PT6, PT4) did not. Participant PT6 and PT11 worked in the same project, although participant PT11 shared results of competitor analysis, product vision, user requirements, customer feedback and usability testing with the development team and participant PT6 was responsive to changes, had a part time usability engineer, gathered requirements from users, conducted user testing and had a shared team ownership of quality and usability, however, he did not consider his model as an AUCDI model due to lack of developers' involvement and presence in user testing. Participant PT6 stated that

> No I cannot say I totally consider it Agile UCD because I needed feedback from users more than anyone else if this could occur the shape of our process could have changed a lot. I really wanted to see users in front on my eyes using the tool and

judge if this tool really succeeded or not.

Participant PT4 did not consider his model as an AUCDI model, he stated that

> No I would not call it an Agile user centric process, it is a regular Agile with some tweaks.

As it can be concluded from the quotes of participants (PT1, PT4, PT6, PT7, PT10, PT12) that there is a lack of clear and widely acceptable road map for AUCDI that could allow participants to evaluate their development model. Venturi and Troost [279] defined *User Centred Design Integration* as follows

> UCD integration is achieved when every phase of the product life cycle follows the principles of user centered design, when UCD team is provided with the proper skills and experience, supported by the management commitment and a proper UCD infrastructure and awareness and culture are properly disseminated in and out of the organisation [279].

Up till now there is an absence of a clear definition of AUCDI that allows the development teams to evaluate their development process.

4.4 Discussion

After the introduction of Agile development processes, Agile teams that attempt to integrate Agile development processes and UCD suffered from the lack of a well established or a widely accepted approach for AUCDI. This was aggravated by facing a set of challenges related to a variety of issues that were discussed in chapter 3 section 3.3. These issues are: lack of allocated time for upfront activities, difficulty of chunking, communication between the development team and UCD practitioners, conducting usability testing, absence of UCD practitioners or their increased work load in case of their presence, and lack of documentation.

The interview study also revealed that in addition to those challenges, Agile teams suffer from the additional challenge of lack of management support to UCD activities that occurred to participants (PT3, PT10, PT14). This was attributed to a variety of reasons, for example, lack of management awareness on UCD impact on the overall quality of the product, lack of awareness on the importance of UCD practitioner role, tight schedules,

and lack of funds. This lack of management support resulted in management reluctance to allocate time, priority or funds for hiring a usability engineer and conducting UCD related activities, for example, usability studies, usability testing, etc.

The study also revealed the lack of awareness of industrial practitioners of the proposed methods in AUCDI literature for achieving the integration, for example, the design studio [277, 278] and Uscrum [262]. The results from section 4.3.1 show that teams are struggling with a variety of integration challenges and attempting their own solutions for dealing with these challenges without benefiting from the proposed solutions in the literature due to lack of awareness of its existence. Two examples illustrate this issue. The first example is shown in section 4.3.1.4 which discusses the challenge of conducting usability testing in Agile teams. The tight Agile time lines, budget constraints, difficulty of accessing users, and lack of full time usability engineers led some participants to conduct usability testing on customers, developers, quality assurance team, or conduct hallway testing rather than testing on users. As a result only 6 projects out of 12 (P2, P3, P6, P10, P11, P12) conducted usability testing on users while the rest of the development teams resolved to testing on development team members, quality assurance team (testers), UCD practitioners, members from other development teams and / or customers. However, literature on integrating Agile and UCD that was discussed in chapter 3, section 3.3.5 offers a number of other alternatives, for example, preparing for user research [136], utilizing discount usability engineering techniques [56, 76, 89, 156, 198], and conducting remote usability testing [59].

The second example for the gap in knowledge transfer between AUCDI literature and industrial practitioners is illustrated in section 8.2.3.3. Section 8.2.3.3 discusses the challenge faced by industrial practitioners as a result of UCD practitioner workload. Table 4.5 reveals that only four teams (P1, P6, P8, P9) included a usability engineer who worked as a part timer. However, the tight Agile time lines and the frequent Agile meetings together with the fact that the usability engineers were a shared resource among a number of projects led to increased work load on usability engineers. Teams resolved the increased UCD practitioner workload by either planning ahead their needs for this shared resources (PT6) or utilizing developers to perform the role of UCD practitioner in case of their absence (PT1, PT9). Literature on integrating Agile and UCD that is discussed in chapter 3, section 3.3.7 offers a number of other alternatives, for example, mentoring process [5], office hours approach, decreasing UCD practitioner workload [76, 181], distributing UCD practitioner workload [283].

These two examples indicate the low awareness of industrial practitioners in regards to the proposed AUCDI literature and the lack of knowledge transfer between AUCDI literature authors and community on one hand and industrial practitioners on the other hand. Industrial practitioners can benefit from a clear road for integrating Agile development processes and UCD that provides roles, activities, and responsibilities involved in the integration. As long as such a road map is absent then industrial practitioners will continue to come up with their own integration approaches or solutions.

This lack of a clear and widely acceptable road map for AUCDI was discussed by participant PT4, a principal technical consultant who had 14 years of total experience in software development and 7 year experience in Agile software development projects and a PhD in information systems. Participant PT4 stated that

> There is no very specific rigid practices for UCD in agility [...] Yes when you talk about building a user centric solution using Agile you will not have a road map, it is a matter of some best practices or techniques but there is no road map [...] there is no team model that create specific roles, there is no standard document templates that describe how you do this and how you describe this there is no communication mechanism and this is very important, there is no modeling technique.

Up till now there is an absence of a clear definition of AUCDI that allows the development teams to evaluate their development process. There is an absence of an integration road map that organizations can follow.

4.5 Limitations

Our study suffered from certain limitations: the majority of data was collected from team members involved in software development related roles, for example, technical team leaders, technical architects, principal technical consultant, and senior software engineers while a smaller set of data was collected from UCD practitioners this was partly due to the fact of lack of UCD practitioners' role in Agile projects and partly due to the availability and willingness of the participant to take part in the study. Better results could have been achieved by interviewing one Agile practitioner and one UCD practitioner from each team.

4.6 Chapter Summary

This chapter investigated industrial practices for integrating Agile development processes and UCD. The emerging themes revealed by the study included a number of AUCDI challenges and methods. The study also revealed the absence of a road map for AUCDI and the existence of a knowledge gap between AUCDI literature and industrial practices. This study findings were compared with findings reported in literature and the results of some of the previous studies were confirmed and others were contradicted.

Chapter 5 will investigate the role that can be played by usability maturity models in the domain of integrating Agile development processes and UCD.

Chapter 5

Agile Development Processes and User Centred Design Integration and Usability Maturity Models

This chapter provides the necessary foundation for chapters 6 and 7. It reports on the research approach followed to investigate the suitability of Usability Maturity Models (UMMs) for utilisation in the context of Agile projects.

5.1 Introduction

In the 1990s a number of UMMs emerged that aimed to assess the organisation's UCD capability and/or performance. UCD capability assessment measures the extent to which UCD is consistently and systematically implemented in the different organisational projects. UCD performance assessment focuses on the extent of effective implementation of UCD in development projects [153]. Jokela et al. [154] stated that the results of this assessment can be used to:

1. **Prioritise areas of improvement.**

   The results of the assessment provide an indication of areas with low maturity rating that require improvements, for example, usability evaluation methods, usability practices documentation.

2. **Identify strengths to be protected.**

3. **Third party certification purposes.**

   The assessment results can be used by a purchaser organisation in comprehending the supplier organisation maturity in developing usable products.

Although Agile and User Centred Design Integration (AUCDI) research is growing, it has not yet exploited the potential of UMMs, which is strongly relevant to AUCDI practice. The SLR content paper classification of AUCDI studies according to the integration approach revealed eight different categories, non of which focused on utilization of UMMs. UMMs can be utilised in the Agile development projects as a diagnostic tool. UMMs can assist in assessing the status quo to evaluate the extent to which UCD is systematically and consistently implemented as well as the extent of effective implementation of UCD in development projects. The results can help organisations identify their strengths and weaknesses in regards to UCD related aspects and accordingly plan for improvement actions.

5.2 Study Aim

This study has two aims:

- To investigate the suitability of UMMs for utilisation in the context of Agile projects in order to assess the organisation's UCD capability and/or performance.
- To investigate the relationship between the success of AUCDI attempts and usability maturity level.

5.3 Research Approach

The research approach that was used to investigate these research questions started by conducting a comparative study of UMMs that are currently available in the public domain, then choosing one or more of those UMMs according to a set of comparative criteria. This was followed by choosing five AUCDI case studies that represented successful and unsuccessful AUCDI attempts. Then the chosen UMM(s) were utilized in assessing the usability maturity level of the five chosen AUCDI case studies. This occurred via conducting a set of one-on-one, Skype interviews. Finally, the results of these studies were synthesised to investigate the following: the existence of a relationship between the success of AUCDI attempts and usability maturity level and the suitability of UMMs for utilisation in assessing usability maturity in the context of Agile projects. This step also involved comparing the results that emerged from assessing the usability maturity level of the five AUCDI case studies via different UMMs.

5.3.1 Comparative Study of Usability Maturity Models

The first step of this research involved conducting a comparative study of UMMs in order to choose one of those UMMs for utilisation in assessing the usability maturity level of case studies that integrated Agile development processes and UCD.

Figure 5.1 highlights the variety of usability maturity/ assessment models that are available in the public domain and shows that the quality grid

of Crosby [49] represents the common ancestor of usability maturity/ assessment models. Usability maturity / assessment models fall into two categories: the first category originate from software process assessment models (Capability Maturity Model (CMM)), and the second category originate from Crosby (QMMG) [49] as cited in [154]. Both Nielsen model and OS-UMM are not shown in this diagram since they emerged in 2006 and 2011 respectively and this diagram emerged in a journal paper in 2006.

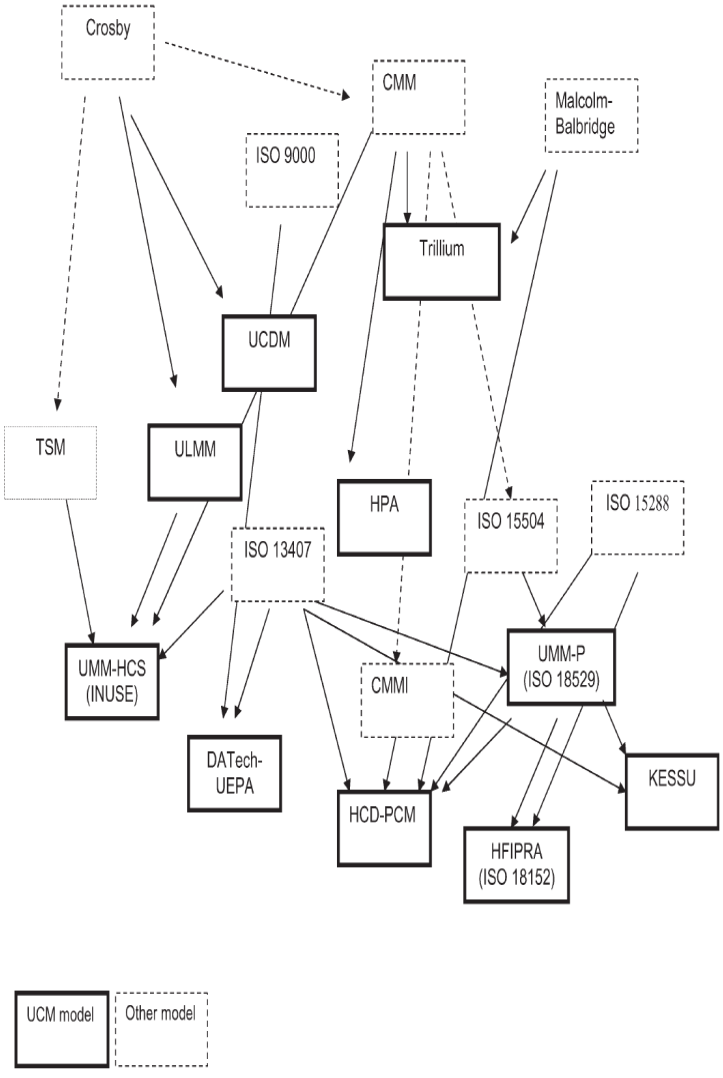

Figure 5.1: Usability Capability Maturity Family tree [154]

To the best of our knowledge there exists eleven usability maturity models for which an overview will be provided below. The features, perspective

of usability integration, maturity approaches, scope, and documentation of these models are diverse. An example of difference in scope is shown in the fact that some models focus on assessing the user centredness of the organisation whereas others focus on measuring the user centredness of projects. Iivari and Jokela [135], Jokela et al. [154] pointed out that the existing UMM literature is focused on presenting the models structure and assessment method rather than the empirical evaluation of these models.

1. **Nielsen Corporate Usability Maturity Model**

   Nielsen Corporate Usability Maturity Model [215] was developed in 2006 by Jackob Nielsen. It is composed of 8 stages or maturity levels as shown in table 5.1. It declares that as organisations' usability matures they progress through the sequence of stages from initial hostility to widespread reliance on user research. It states that the sequence of maturity stages is fairly universal and consequently any organisation can be matched with this maturity stages description to see what the next stage is likely to be [215].

2. **Usability Maturity Model-Human Centredness Scale (UMM-HCS)**

   UMM-HCS [68] was developed in 1998 by the European INUSE project that focused on the assurance of interactive systems or web sites usability. UMM-HCS was derived from all significant existing UMMs as shown in figure 5.1. It embodies usability maturity via 6 maturity levels of a combination of attitude, technology, and management activities. It offers organisations an understanding of how organisation's usability maturity progress and allows organisations to measure their maturity and subsequently plan for improvement. UMM-HCS documentation has an assessment recording form and its use is described [68].

3. **Trillium**

   Trillium [40] is a process assessment approach for development of telecommunication products, developed in 1991 by Bell Canada. Trillium Model is based on (CMM) version 1.1 as shown in figure 5.1. Although the model is extensively documented and published in the public domain, yet, it does not discuss methods on how to perform the assessment.

4. **Usability Leadership Maturity Model (ULMM)**

   IBM developed Usability Leadership Maturity Model (ULMM) [87, 244] in the early 1990s to benchmark and improve the usability status

in software projects. It examines organisations from three different aspects: skills, organisation, and processes. The model documentation that is available in the public domain is very limited and only two papers are available [87, 244].

5. **Humanware Process Improvement (HPI)**

   Humanware Process Improvement was developed by Philips [101, 192]. HPI has 10 Key Process Areas (KPAs) that pinpoint practices for UI design and suggest the method to integrate these practices in the process of product creation [101]. HPI embraces the cycle of "Plan-Do-Check-Act" and covers software and hardware aspects of products [192]. HPI provides a questionnaire as an assessment tool for self and full assessments [101].

6. **User Centred Design Maturity (UCDM)**

   The University of Loughborough developed User Centred Design Maturity (UCDM) model [70] as a tool to benchmark the capability of information systems in the UK public sector. UCDM has five capability areas, for example, "systems design", "project structure and goals" and maturity stages, and 15 foci of assessment [154].

7. **Usability Maturity Model-Processes (UMM-P)**

   The European INUSE project developed UMM-P [69] to be utilised in the assessment and improvement of the human centred processes in system development. The model includes seven processes defined by a number of practices and work products. Examples of UMM-P processes are "Plan and manage the human-centred design process", and "Specify the user and organisational requirements". UMM-P includes and describes an assessment recording form.

8. **KESSU**

   Oulu University in Finland developed KESSU [142, 143, 144, 145, 146, 147, 149, 150] as an assessment model. The model resulted from a number of empirical studies that utilised (UMM-P+ISO 15504) and concluded that they were not appropriate for usage in industrial context. KESUU focuses on examining the performance of user centred processes rather than the managerial aspects. KESSU has seven processes of UCD. These processes are identify user, context of use, determine user requirements, produce user task designs, produce interaction designs, usability feedback and usability verification [154].

9. **Procedures for Usability Engineering Process Assessment (DATech-UEPA)**

   DATech-UEPA was developed in Germany and has usability engineering as the main focus of assessment. The model identifies three maturity levels and 19 assessment foci. The model is tailored to assess the manufacturing organisations and is documented in German as cited in [154].

10. **Human Centred Design-Process Capability Model (HCD-PCM Design and Visioning)**

    The HCD-PCM model was developed by Mitsubishi Research Institute, NTT Advanced Technology and Otaru University of Commerce in 2002. The model has a wide scope over the system development life cycle. It includes various processes from market visioning to system disposal process capability types. HCD-PCM model is composed of two models: one for design processes and the other for visioning process. The capability level for each of the visioning and the design process identifies five capability levels [154].

11. **Open Source Usability Maturity Model (OS-UMM)**

    OS-UMM [245] is a UMM for open source projects. It was developed by members from University of Western Ontario, Canada and Faculty of Information Technology, United Arab Emirates University. The OS-UMM model has five maturity levels. The proposed maturity scale has key usability factors, such as user requirements, usability learning, user centred design methodologies, learnability, attractiveness, usability bug reporting, and usability testing [245].

Table 5.1 provides a comparison for the maturity levels of the different UMMs. It shows that the different UMMs vary in the number of maturity levels between eight (Nielsen Model) and three (DATech-UEPA).

The table shows that the different models can use different terms for referring to maturity levels that cover same aspects. For example, Systematic usability process (Nielsen), institutionalized (UMM-HCS, OS-UMM) and fully integrated (Trillium) as a reference to fully integrated UCD activities along the development life cycle throughout all projects. Another example is continuous improvement (DATech-UEPA), Optimize (HCD-PCM Design) and optimizing (UMM-P) as a reference to continuous organizational efforts to improve UCD across all projects.

Nielsen	UMM-HCS	Trillium	UCDM	UMM-P	DATech-UEPA	HCD-PCM Design	HCD-PCM Visioning	OS-UMM
Hostility toward usability	Unrecognised	Unstructured	Uncertainty	Incomplete	Introduced	Do	Awake	Preliminary
Developer-centred usability	Recognised	Repeatable	Awakening	Performed	Reproducible Results	Plan	Know	Recognised
Skunk works usability	Considered	Defined	Enlightenment	Managed	Continuous Improvement	Control	Understand	Defined
Dedicated usability budget	Implemented	Managed	Wisdom	Established	–	Adapt	Infer	Streamlined
Managed usability	Integrated	Fully Integrated	Certainty	Predictable	–	Optimize	Create	Institutionalized
Systematic usability process	Institutionalized	–	–	Optimizing	–	–	–	–
Integrated user-centred design	–	–	–	–	–	–	–	–
User-driven corporation	–	–	–	–	–	–	–	–

Table 5.1: Maturity Levels of Usability Maturity Models

Criteria for Comparing Usability Maturity Models

The following criteria were found relevant for the purposes of this work in order to compare the main characteristics of the different UMMs that are available in the public domain.

Lightweight

Since the chosen UMM will be utilized to assess the usability maturity level of each project, the model should be lightweight; it should require low overhead so as not to disrupt any Agile project schedule and low cost so as not to consume considerable time to perform the assessment or require additional personnel to conduct the assessment.

Detailed English Documentation

The model should provide detailed documentation that provides practitioners with detailed maturity model definition and detailed descriptions of the assessment process. This detailed documentation will provide explicit guidance to practitioners to conduct self assessment, i.e., conduct the assessment on their own without the need for the presence of the model author. Since the available usability maturity/assessment models were developed in various countries, one of the criteria for choosing the usability maturity/assessment model is to be documented in English.

Domain Independent

Some UMMs are domain specific; i.e., they were created for specific domains like telecommunication or manufacturing. This imply that they focus on domain specific practices and cannot be utilized in other domains. Thus the model chosen should be domain independent; i.e., it should be suitable for utilisation in all organisations regardless of their domain of business.

Empirically Evaluated

The model should have been evaluated in empirical studies and iterated according to the results of these empirical evaluations.

The UMMs were examined using the criteria identified above and table 5.2 presents a summary of the results of the comparison between the different usability models according to the comparison criteria proposed earlier.

The UMM pursued is documented in English, has a detailed documentation, can be used irrespective of the organisation's domain (generic), is lightweight, and has been empirically evaluated.

These criteria led to the exclusion of a number of UMMs.

The language criteria led to the exclusion of DATech-UEPA, HCD-PCM Design and HCD-PCM Visioning since they were not documented in English. The detailed documentation criteria led to the exclusion of ULMM, HPI, UCDM, KESSU, and OS-UMM since they do not provide sufficient documentation to conduct the assessment in a self assessment form. However, this does not imply that they did not have extensive documentation, for example, KESSU is published in numerous papers [142, 143, 144, 145, 146, 147, 149, 150], however, all these papers and documentation provided a rather high level detail on the dimensions and processes rather than on the how to conduct the assessment. The domain criteria led to the ex-

UMM Model	Language	Detailed Documentation	Domain	Lightweight	Empirical Evaluation
Nielsen	English	Yes	Generic	Yes	No
UMM-HCS	English	Yes	Generic	Yes	Yes
Trillium	English	Yes	Telecommunication	No	Not published
ULMM	English	No	Generic	N/A	Not Published
HPI	English	No	Consumer Product Development	N/A	Not Published
UCDM	English	No	Information System Capability in UK Public Sector	N/A	Not Published
UMM-P	English	Yes	Generic	No	Yes
KESSU	English	No	Generic	N/A	Yes
DATech-UEPA	German	N/A	Manufacturing	N/A	N/A
HCD-PCM Design	Japanese	N/A	Generic	N/A	N/A
HCD-PCM Visioning	Japanese	N/A	Generic	N/A	N/A
OS-UMM	English	No	Open Source Projects	N/A	N/A

Table 5.2: Criteria for Choosing a Usability Maturity Model

clusion of Trillium since it is domain specific as it is focused on the telecommunication industry. The lightweight criteria led to the exclusion of UMM-P since the model is not lightweight and requires considerable time to be conducted which may disrupt any Agile project schedule and impose considerable cost as a result of consuming considerable time to perform the assessment.

In case of the lack of enough documentation or the lack of English documentation, it was hard to judge whether the model is lightweight or not. As a result "Not Applicable (N/A)" was used as an indication that it was not applicable to make an evaluation.

This left us with only two UMMs: Nielsen Corporate Usability Maturity Model and Usability Maturity Model-Human Centrdness Scale. Although Nielsen model was not empirically evaluated, yet it was decided to utilise both models in five AUCDI case studies in order to provide richer comparative analysis via two UMMs.

Comparison of Nielsen Model and UMM-HCS Model

The Nielsen model and UMM-HCS model were compared in regards to model design and structure and the results of this comparison are reported in table 5.3.

Criteria	Nielsen Model	UMM-HCS
Purpose of use	Descriptive	Descriptive
Number of Practices	24	32
Scoring Scheme	Yes/No	Not achieved Partially achieved Largely achieved Fully achieved
Empirical Evaluation	None	Developed iteratively based on trials

Table 5.3: Comparing Nielsen and UMM-HCS Models

Purpose of Use

Chapter 2 discussed the difference between descriptive and prescriptive maturity models. Descriptive maturity models are used as a diagnostic tool [186] to assess the current capabilities of the examined entity against specific criteria [16], prescriptive maturity models on the other hand aim to pinpoint desirable maturity levels and provide specific and detailed improvement guidelines [16, 186]. According to this definition both Nielsen model and UMM-HCS are descriptive models. Although UMM-HCS documentation claims that the model assists organisations in improving their maturity level, yet there is an absence of guidelines to identify the desirable maturity levels and the desired measures for improvement.

Number of Practices

Nielsen Model has 24 practices and UMM-HCS has 32 practices.

Scoring Scheme

The scoring scheme of Nielsen model depended on identifying the achievement or non achievement of particular practices while UMM-HCS model scoring scheme required the evaluation of the satisfaction of each practice by one of these values: not achieved, partially achieved, largely achieved and fully achieved.

Empirical Evaluation

Nielsen model did not report on results of empirical evaluation, whereas, Usability Maturity Model (UMM) [67] was evaluated in three trial assessments [67, 170] and the structure was perceived as too complex to be understandable in practice. UMM-HCS [68] and UMM-P [69] refined the structure. Although UMM-HCS was empirically evaluated by its author

yet, it was not sufficiently empirically validated in self assessment endeavours.

5.3.2 Choosing Five AUCDI Case Studies

The second step of the research approach involved choosing five case studies that integrated Agile development processes and user centred design and represented successful and unsuccessful AUCDI attempts

A candidate list of five academic researchers and industrial practitioners were selected. This list included authors who developed new AUCDI approaches and whose work on AUCDI was well received and highly referenced. The chosen case studies also reflect a "two tail" design [286] in which cases from both extremes (success and failure) are selected.

Interviewees were contacted via an email. All of the interviewees were professional usability practitioners whose job roles were: usability product manager, usability analyst, usability engineer, lead user experience designer, and team manager for user experience design. One interview was conducted per participant in which questions regarding Nielsen model and UMM-HCS were posed. At the beginning of the interview the interviewees were asked for their permission to record the interview and then the informed consent form (Appendix H) was read and their consent on its terms was recorded. All interviewees agreed to record the interview. All interviews were transcribed in English. After transcribing the interviews some of the participants were recontacted with some follow up questions for clarifying incomplete or ambiguous answers to some questions. Table 5.4 provides a summary of the different case studies

Case Study	**Product**	**Agile Method**	**AUCDI Status**
Case Study 1	Ecommerce platform	Scrum	Successful
Case Study 2	Approval process editor	Scrum	Successful
Case Study 3	Touch screen kiosk system for mechanics	Scrum	Successful
Case Study 4	TV show planning software	Scrum	Successful
Case Study 5	Online purchase and service management software	Scrum	Unsuccessful

Table 5.4: AUCDI Case Studies

The table suggests that each case study works in a different sort of software. All case studies implement Scrum. Four case studies represent successful integration attempts and one case study represent a failure integration attempt. The imbalance between the number of successful and unsuccessful case studies is attributed to the existence of publication bias in the AUCDI domain as indicated by the results of the SLR that was reported in chapter 3. Publication bias occurs when more positive results are published than negative results [162]. The SLR revealed that only one paper reported failure in AUCDI literature and 46 other papers reported success. For the sake of protecting the anonymity of participants, company names and some details on the projects will be withheld.

Case Study 1

The product developed in the first case study was a Software as a Service where high quality UX was crucial. The product developed an integration with a new ecommerce platform in order to allow sellers to sell their inventory to multiple ecommerce sites. The interviewee worked as a usability product manager, and was responsible for managing all the UX aspects of the project and designing the user interfaces. This project represented successful AUCDI attempt as declared by the author. This success occurred through utilising a variation of Scrum. The project had two peer product owners, one product owner focused on functionality and the other focused on usability and user experience. The usability product owner established the software's UX vision along with a set of personas. The UX vision was embodied in high level user goals, high level UX description of the product and a high level navigation model for the product.

Case Study 2

The second case study occurred at a company working in the domain of enterprise customer relationship management.

The product developed was a second version of an approval process editor, the first version had very poor UX and users could not use the product as expected and there was poor adoption. Four years later it was subject to redesign.

The interviewee worked as a usability analyst which involves being a dedicated UX researcher. He was providing support for around 25 different Scrum teams with research.

This project represented successful AUCDI attempt as declared by the author. This occurred via the R & D team and management team who first,

introduced a new resource plan that reduced workload on UX team members and introduced the concept of office hours in which UX team members would dedicate two hours per week to assist Scrum teams with no UX resources assigned. Second, they introduced design transformations via utilising parallel tracks, one release ahead, communicating designs to developers, interactive prototypes for usability testing, and design studios. Third, using Rapid Iterative Test and Evaluation (RITE) for usability testing in all Agile projects.

Case Study 3

The third case study occurred at a home shopping network company. The product developed was a new system that facilitates planning of shows for a home shopping network. Show planners used the software in order to schedule shows, products to shows and hosts to shows. The interviewee worked as a lead UX designer, running a team with three designers. This case study represented successful AUCDI attempt as declared by the author. This success occurred through the use of a rapid process that allows the collaboration of developers, designers and stakeholders in exploring design alternatives. This occurred via providing participants with common guidelines that are utilised in producing several sketch designs, then participants took part in a collaborative workshop to discuss each other's work. Then participants ideas were merged to one design concept that was taken forward. This process reportedly had several advantages, for example, facilitating role sharing and knowledge transfer, allowing rapid exploration of design alternatives, and fostering a shared understanding of design vision.

Case Study 4

The fourth case study occurred at a company working in the domain of asset performance management. The product developed was a touch screen kiosk system for mechanics in order to receive work orders and record the results of the repair. The interviewee worked as a usability engineer. This case study represented successful AUCDI attempt as declared by the author. This success occurred through the use of a development approach that is based on extreme programming and scenario based design. This approach uses parallel tracks and allows the usability engineer to work one iteration ahead of the developers. The key design is represented in a set of mock-ups, scenarios, claims, and design goals that support synchronization activities and help the usability engineer plan and execute usability evaluations.

Case Study 5

The fifth case study occurred at an Internet infrastructure services company that works in domain name resolution and registration as well as protecting and enabling interactions across the world's voice, video, and data networks. The product developed was a redesign of a web site for purchasing and managing services on-line. The project goal was to boost sales, retain customers, and reduce the support calls amount via an improved UX of the purchase and product renewal processes. The interviewee worked as a manager of UX design team which was a central team serving the whole company. This case study represented unsuccessful AUCDI attempt as stated by the author. At the end of the release the project went into the beta phase and a major problem was discovered since the users were unable to complete their purchase successfully. The product failed in improving customer retention and increasing sales. This resulted in pulling the software back in order to fix it. The cause of this failure was attributed to a lack of cross functional team members communication due to geographical separation and reluctance of the engineering team to collaborate with non engineering teams and as a result the product manager and the UX team were prevented from participating in the sprint planning and Scrums. This led the engineering team to interpret and implement the designs incorrectly. There was also a lack of iterative refinement of designs.

5.3.3 Mapping The Practices of Nielsen Model and UMM-HCS

Mapping the practices of the chosen UMMs was particularly important since due to the tight schedules of the interviewees, it was only possible to conduct one interview per interviewee. Thus questions were posed regarding both Nielsen Model and UMM-HCS in the same interview. As a result the documentation of both models were carefully examined in order to check commonalities and differences between practices and consequently avoid redundancy of posing questions that aim to evaluate the same practice.

Table 5.5 provides a mapping between the practices of both models.

Table 5.5 shows that all aspects of Nielsen model are subsumed within UMM-HCS, i.e., it covers all aspects addressed by Nielsen model in addition to further aspects. For example, Nielsen model covers the finan-

Nielsen Practice ID	Practice	UMM- HCS Practice ID
A1.1	Recognition of usability importance	A1.1, B1.1
A1.2	Recognition of importance of understanding user needs	A1.1, B2.1, B2.2
A1.3	Developers not acting on behalf of users	B2.1, B2.2
B1.1	Recognition of usability importance	A1.1
B1.2	Willingness to allocate funds for usability activities	Implicit
B1.3	Allocating funds for usability activities	Implicit
B1.4	Presence of dedicated funds for usability activities	Implicit
C1.1	Presence of internal usability practitioner	Implicit (C3.1, C3.2, C3.3)
C1.2	Presence of a usability team led by a usability manager	Implicit (C3.1, C3.2, C3.3)
D1.1	Performing usability testing	C1.1, C1.2, C1.4, D2.1, D2.2, D2.3
D1.2	Planning for usability	C1.1-C1.4, C2.1-C2.3, C3.1-C3.3, D1.1-D1.3, D2.1-D2.3, D3.1, D3.3
D1.3	Presence of a dedicated usability lab	C2.2
D1.4	Utilising a usability reports archive to compile past usability findings	D.3.2
D1.5	Performing early user research	A2.1, A2.2
D1.6	Performing iterative design	D2.1, D2.2, D3.1
D1.7	Quantitative usability metrics can be used to track quality	C1.3
D1.8	Projects has defined usability goals	C1.3
E1.1	Presence of a tracking process for user experience quality throughout design projects and across releases	D1.1
E1.2	Utilising user interface design standards or a centralised definition of preferred design patterns	E1.1-E1.3, E2.1-E2.2
E1.3	Projects are prioritised according to the business value of their user experience	E1.1-E1.3, E2.1-E2.2
E1.4	Recognition of the need for user centred design process	D1.1, E2.1
E1.5	User research data is employed to determine individual projects to be built	E1.1-E1.3, E2.1-E2.2
E1.6	Concept of total user experience extend to other forms of customer interaction with the company	B1.3, E1.1, E1.2, E1.3
E1.7	User research data is employed to determine the type of projects to fund	E1.1, E1.2, E1.3

Table 5.5: Mapping Between Practices of Nielsen Model and UMM-HCS

cial usability aspects through three practices B1.2, B1.3, and B1.4. These three practices are an important factor for the transition of maturity level three to maturity level four. Nevertheless, UMM-HCS does not have any management practices that discuss fund approval or allocation. Yet it is implicitly indicated that the funding is allocated since UMM-HCS has a number of practices that require significant funding. Those practices are B1.1, B1.2, B1.3, B2.1, B2.2, C.2.2, C2.3, and C3.2. Moreover, practice D1.2 in Nielsen model which is concerned with usability planning is covered thoroughly in a number of management practices in UMM-HCS that represent maturity level four or C (implemented) and level five or D (integrated). These practices are: C1.1-C1.4, C2.1-C2.3, C3.1-C3.3, D1.1-D1.3, D2.1-D2.3, D3.1, and D3.3. Finally, practices E1.2 and E1.5 in Nielsen model which is concerned with "interface design standard or a centralised definition of preferred design pattern and user research data is used to determine individual projects to be built" is covered implicitly in UMM-HCS in a number of management practices that represent maturity level 6 or E (institutionalized). Those practices are E1.1-E1.3, and E2.1-E2.2.

Method of Posing Interview Questions

In case of the presence of commonalities between UMM-HCS and Nielsen model practices one question was posed to cover both practices. An example that illustrates this issue is practice D1.3-Presence of a dedicated usability lab in Nielsen Model and practice C2.2-Provide suitable facilities and tools in UMM-HCS. Since practice C2.2 asks about both facilities and tools, thus a question was posed on the available facilities and tools and the interviewee was encouraged via prompts and probes to elaborate on the existing facilities and tools. If the interviewee happens to mention dedicated usability lab among the facilities available then answer is acquired for Nielsen model as well, if not then a question was specifically posed to check the presence of a dedicated usability lab.

A second example is practice D1.1- Performing usability testing in Nielsen Model and practice C1.4- Continuous Evaluation in UMM-HCS, those practices are concerned with usability testing. Nielsen model is concerned about usability testing performance and UMM-HCS is concerned about the frequency of its performance, thus a question was posed on the performance of usability testing in general and then the interviewee was encouraged to elaborate more on the frequency of conducting usability testing.

A third example is Practice D1.6- Performing iterative design in Nielsen model and Practice D2.2- Change based on Feedback in UMM-HCS both

practices aimed to evaluate the acquisition of feedback that led to iterative design. Thus a question was posed on acquiring design feedback and how it affects design.

Although this approach allowed for saving the interviewee's time and effort via avoiding redundant questions yet the downside of it was that in chapters 6 and 7 some quotes were used for practices in both Nielsen model and UMM-HCS.

Table 5.6 provides the list of practices that used common quotes.

Practice Number in Nielsen	Practice Number in UMM-HCS
D1.1	C1.4
D1.3	C2.2
D1.4	D3.2
D1.5	A2.1
D1.6	D2.2
D1.8	C1.3

Table 5.6: Common Quotes between Nielsen Model and UMM-HCS

5.3.4 Utilising The Chosen UMM(s) in five AUCDI Case Studies

The fourth step of the research approach involved utilising the chosen UMM(s)(Nielsen Model and UMM-HCS) in assessing the usability maturity level of five case studies that integrated Agile development processes and user centred design and represented successful and unsuccessful AUCDI attempts. This occurred via conducting a set of one-to-one, Skype interviews.

A set of open ended questions were formulated and posed to interviewees to evaluate the achievement of different practices. The questions were formulated in a manner that is precise, unambiguous, and understandable to respondents. It was decided not to pursue a Yes/ No answers as implied by Nielsen model or Not Achieved, Partially Achieved, Largely Achieved or Fully Achieved answers as indicated by UMM-HCS model but rather to use open ended questions. This was attributed to three reasons. First, evidence needed to be acquired that support the interviewee's answers. Second, Yes/ No questions can be problematic as they suffer from acquiescence bias, problems with lack of reliability since participants provide

different answers on different instances and imprecision since they restrict measurement to only two values [163]. Third, open-ended questions have the advantage of allowing in-depth clarification, enabling testing of the respondents' knowledge limit, reaching a truer assessment of respondent's knowledge, and producing unanticipated answers that provide illuminating information [247].

Probes and prompts [247] were also used. Probes were used to get the interviewee to elaborate on a response. This occured via obvious tactics, such as asking "Anything more?" or "Could you go over that again?" Also if the answer were very generic, a probe was used to acquire a personal response, e.g. "what is your personal view on this?" Other useful tactics were used, such as the use of silence period, mmhmm, or repeating back all or part of the interviewee answer. Prompts involve the interviewer offering a range of possible answers to the interviewee and must be used consistently in all interviews [247].

Answers to interview questions were used in evaluating the usability maturity level of each case study and the results of each case study were compared with the achieved practices in the different usability maturity levels in order to determine the closest usability maturity level. Those results are reported in chapter 6 in table 6.2 and in chapter 7 in tables 7.5, 7.6, 7.7, 7.8, and 7.9.

5.3.5 Synthesizing The Results Of Utilisation of UMMs

The fifth step of the research approach involved synthesizing the results of utilisation of Nielsen Model and UMM-HCS in order to investigate the following: the existence of a relationship between the success of AUCDI attempts and usability maturity level and the suitability of UMMs for utilisation in assessing usability maturity in the context of Agile projects. This step also involved comparing the results that emerged from assessing the usability maturity level of five AUCDI case studies via different Nielsen Model and UMM-HCS.

The results of this step are reported in chapters 6 and 7.

5.4 Chapter Summary

This chapter reported on the research approach followed to investigate the suitability of Usability Maturity Models (UMMs) for utilisation in the context of Agile projects. The following chapter will report on applying Nielsen model on five case studies that integrated Agile development processes and UCD and utilising the model in assessing their usability maturity level.

Chapter 6

An Empirical Study of Nielsen Corporate Usability Maturity Model

This chapter investigates the suitability of the Nielsen Usability Maturity Model for utilisation in the context of Agile projects in order to assess the organisation's UCD capability and/or performance. It reports on the utilisation of Nielsen Model in assessing the usability maturity level of five AUCDI case studies and investigating the presence of a relationship between the success of AUCDI and usability maturity levels. An analysis and general observations are provided for the suitability of Nielsen model in the context of Agile projects. This chapter offers a critique to the model from a self assessment perspective. It ends by listing the research challenges faced and the research limitations.

6.1 Utilising Nielsen Corporate Usability Maturity Model in AUCDI Case Studies

This section discusses the steps that were used in order to utilise Nielsen model in assessing the usability maturity level of the five AUCDI case studies introduced in chapter 5.

Maturity models as a design product can take various forms: pure textual description, functioning of the maturity model, or instantiation as a software assessment tool [201]. The Nielsen model is a textual model that is composed of 8 maturity levels but is written primarily as a textual narrative and this form cannot be easily deployed for measuring usability maturity of an organisation. Accordingly, the model was carefully examined in order to transform its narrative form into a set of measurable dimensions and practices. Each dimension is composed of a number of practices. This examination resulted in a model that is composed of five dimensions and 24 practices as shown in table 6.1. These dimensions are: developers' attitude towards usability, management attitude towards usability, usability practitioners' role, usability methods and techniques, and strategic usability. Nielsen model describes each usability maturity level as a set of achieved practices. Thus 1 was used to signify an achieved practice and 0 was used to signify a non achieved practice in each usability maturity level. An identifier were given to each practice in order to assist with the mapping of this model and UMM-HCS model that is reported in chapter 7.

Table 6.1 shows the dimensions, practices, and maturity levels involved in Nielsen model.

6.2 Results

This section reports on applying Nielsen Corporate Usability Maturity Model in the five AUCDI case studies that were discussed in chapter 5 and utilising the model in assessing their usability maturity level.

Interviewees from case study 1, 2, 3, 4, 5 are referred to as PT1, PT2, PT3, PT4, and PT5 respectively. In order to protect the anonymity of participants, the word "he" will be used to refer to all of them. Relevant interviewee quotes are included to illustrate the evaluations given to some

Dimension	Practices	ID	1	2	3	4	5	6	7	8
Developers Attitude Towards Usability	Recognition of usability importance	A1.1	0	1	1	1	1	1	1	1
	Recognition of importance of understanding user needs	A1.2	0	1	1	1	1	1	1	1
	Developers not acting on behalf of users	A1.3	0	0	1	1	1	1	1	1
Management Attitude Towards Usability	Recognition of usability importance	B1.1	0	1	1	1	1	1	1	1
	Willingness to allocate funds for usability activities	B1.2	0	0	1	1	1	1	1	1
	Allocating funds for usability activities	B1.3	0	0	1	1	1	1	1	1
	Presence of dedicated funds for usability activities	B1.4	0	0	0	1	1	1	1	1
Usability Practitioners Role	Presence of internal usability practitioner	C1.1	0	0	0	1	1	1	1	1
	Presence of a usability team led by a usability manager	C1.2	0	0	0	0	1	1	1	1
Usability Methods and Techniques	Performing usability testing	D1.1	0	0	1	1	1	1	1	1
	Planning for usability	D1.2	0	0	0	1	1	1	1	1
	Presence of a dedicated usability lab	D1.3	0	0	0	0	1	1	1	1
	Utilising a usability reports archive to compile past usability findings	D1.4	0	0	0	0	1	1	1	1
	Performing early user research	D1.5	0	0	0	0	0	1	1	1
	Performing iterative design	D1.6	0	0	0	0	0	1	1	1
	Quantitative usability metrics can be used to track quality	D1.7	0	0	0	0	0	0	1	1
	Projects has defined usability goals	D1.8	0	0	0	0	0	0	1	1
Strategic Usability	Presence of a tracking process for user experience quality throughout design projects and across releases	E1.1	0	0	0	0	0	1	1	1
	Utilising user interface design standards or a centralised definition of preferred design patterns	E1.2	0	0	0	0	0	1	1	1
	Projects are prioritised according to the business value of their user experience	E1.3	0	0	0	0	0	1	1	1
	Recognition of the need for user centred design process	E1.4	0	0	0	0	0	1	1	1
	Usability data is employed to determine individual projects to be built	E1.5	0	0	0	0	0	0	1	1
	Concept of total user experience extend to other forms of customer interaction with the company	E1.6	0	0	0	0	0	0	0	1
	User research data is employed to determine overall direction and priorities	E1.7	0	0	0	0	0	0	0	1

Table 6.1: Nielsen Corporate Usability Maturity Model Levels [215]

of the assessed practices and these quotes are identified by the participant who said them. Case study 1, 2, 3, 4 and 5 will be referred to as CS1, CS2, CS3, CS4, and CS5 respectively.

Table 6.2 shows the results of applying Nielsen model on CS1, CS2, CS3, CS4, and CS5. The following section discusses the findings from applying

Dimension	Practices	ID	CS1	CS2	CS3	CS4	CS5
Developers Attitude Towards Usability	Recognition of usability importance	A1.1	1	1	1	1	0
	Recognition of importance of understanding user needs	A1.2	1	1	1	1	0
	Developers not acting on behalf of users	A1.3	1	1	1	1	0
Management Attitude Towards Usability	Recognition of usability importance	B1.1	1	1	1	1	1
	Willingness to allocate funds for usability activities	B1.2	1	1	1	1	1
	Allocating funds for usability activities	B1.3	1	1	1	1	1
	Presence of dedicated funds for usability activities	B1.4	1	1	0	1	1
Usability Practitioners Role	Presence of internal usability practitioner	C1.1	1	1	1	1	1
	Presence of a usability team led by a usability manager	C1.2	0	1	1	0	1
Usability Methods and Techniques	Performing usability testing	D1.1	1	1	1	1	0
	Planning for usability	D1.2	1	1	1	1	1
	Presence of a dedicated usability lab	D1.3	0	1	0	0	1
	Utilising a usability reports archive to compile past usability findings	D1.4	1	1	1	1	1
	Performing early user research	D1.5	1	1	1	1	1
	Performing iterative design	D1.6	1	1	1	1	0
	Quantitative usability metrics can be used to track quality	D1.7	1	0	1	1	0
	Projects has defined usability goals	D1.8	1	1	1	1	0
Strategic Usability	Presence of a tracking process for user experience quality throughout design projects and across releases	E1.1	0	1	0	0	0
	Utilising user interface design standards or a centralised definition of preferred design patterns	E1.2	1	1	1	1	1
	Projects are prioritised according to the business value of their user experience	E1.3	0	?	0	0	0
	Recognition of the need for user centred design process	E1.4	1	1	1	1	1
	Usability data is employed to determine individual projects to be built	E1.5	1	1	1	1	1
	Concept of total user experience extend to other forms of customer interaction with the company	E1.6	0	1	0	1	0
	User research data is employed to determine overall direction and priorities	E1.7	0	1	1	1	0

Table 6.2: Case Study

Nielsen Corporate Usability Maturity Model in five AUCDI case studies.

6.2.1 Rating of Nielsen Practices

Table 6.2 reflects the results of maturity level evaluation of Nielsen model for CS1, CS2, CS3, CS4, and CS5.

Dimension 1: Developers Attitude Towards Usability

Practice A1.1- Recognition of usability importance

The rating for practice A1.1 for CS1, CS2, CS3, and CS4 is "Yes" and "No" for CS5. In four case studies developers showed awareness and support in regards to usability. Developers support to usability had two explanations: first, their inability to produce professional designs or conduct usability testing and their awareness that usability practitioners are qualified to perform this role. Second, their desire to produce a software with improved usability in order to achieve higher user satisfaction and avoid rework.

Participant PT1 discussed the reasons behind the positive attitude of developers towards usability by stating that

> The developers were really very good about it [usability] and I think that from the developers point of view what has happened previously is that they would do something [software] and it would not be good enough and they would have to go back and change it and then change it again, right, and still no one was happy because the end product was still not usable so they were actually feeling that they spend so much effort on it and they were still, no one was saying very good developers. You guys came with a great product they actually, they were more on board.

Participant PT5, expressed the reasons for lack of understanding and support of the development team to usability activities by declaring that

> The engineering team were used to work sort of on their own and pretty much the projects were designed by engineers so at the time they have not worked with the UX team before so it was a new thing for them so I do not think they really understood what our purpose was and how we would actually improve the products.

Moreover, participants discussed continuous efforts exerted to raise and maintain awareness on usability via conducting formal and informal us-

ability training as follows:

Participant PT3 discussed the cooperation of developers in attending UCD training by stating that

> I have probably done 20 of them [design studio training] and every single one has been fully attended.

In CS1 and CS3 formal usability training sessions were conducted to staff via the usability product owner in CS1 and lead UX designer in CS3 to raise awareness on usability. Moreover, in CS3 staff were asked to read and discuss particular parts of usability books in order to raise usability awareness.

Participant PT1 discussed staff interest to attend usability training by declaring that

> There was a whole lot of interest and people realised that there was something to learn from this perspective.

Participant PT3 discussed his method in educating staff on usability via reading particular books by stating that

> When I start working with a new team [...] I would have them to read the first chapter of "don't make me think".

Participant PT4 described informal usability training conducted by declaring that

> I have done like a workshop type thing in the company what we call a brown bag lunches [...] I will give you an introduction to usability and what we are doing [...]

Practice A1.2- Recognition of importance of understanding user needs

The rating for practice A1.2 for CS1, CS2, CS3, and CS4 is "Yes" and "No" for CS5. Participant PT1 quote on practice A1.2 indicates the developers' recognition of the importance of understanding user needs since they have witnessed the consequences of rework as a result of lack of user satisfaction. Moreover, participant PT1 raised awareness regarding the importance of understanding user needs via conducting training to staff that covered conceptual modeling, user mental modeling, and tasks modeling.

Participant PT2 discussed the reasons that motivated developers to support usability by stating that

> There is no precedent to this type of tool but they definitely wanted us to prototype and test it and get it right before they jumped in and started developing.

Practice A1.3- Developers not acting on behalf of users

The rating for practice A1.3 for CS1, CS2, CS3, and CS4 is "Yes " and "No" for CS5 since developers acted on behalf of users.

Participants conducted formal and informal training sessions to staff in regards to user consideration and context of use training. User consideration training focused on raising awareness to consider the needs of end users when developing or maintaining the system. Context of use training focused on raising staff awareness to the difference between the skills, motivation and background of end users from developers or support staff.

In CS3 monthly training sessions were conducted by lead UX designer to staff and personas were used to raise staff awareness to the importance of considering the needs of end users when developing the system. In CS4 and CS5 informal training was conducted that discussed user consideration.

Participant PT3 discussed the details of conducting context of use training to staff by declaring that

> I always address that early on, yes I did present and show kind of the triangle, if you will, of business needs, user needs and technical needs.

Dimension 2: Management Attitude Towards Usability

Practice B1.1- Recognition of usability importance

The rating for practice B1.1 for CS1, CS2, CS3, CS4, and CS5 is "Yes". Management support to usability aspects was evident via a number of issues: allocating funds for usability activities (CS1, CS2, CS3, CS4, CS5), hiring qualified UCD professionals (CS1, CS2, CS3, CS4, CS5), conducting training on UCD related aspects to staff (CS1, CS2, CS3), close customer collaboration (CS4), close user collaboration (CS1, CS2, CS3, CS4). In addition for CS1, a lead UCD professional (usability product owner) was hired and offered a a high organisational position that is equivalent to that of the product manager in order to ensure that usability issues will not take less priority than functional issues. Whereas for CS2 a UX team was set as a centralised group that is assigned to the different projects. Management recognition of usability importance was embodied via different methods,

for example, providing time and resources for conducting UCD activities and trusting UCD practitioners with usability and/or UX decision making.

Participant PT1 discussed reasons that led management to recognise usability importance by stating that

> The products they had up to that point, the biggest issue with it was usability, users were complaining and it was to the point that the company, you know the management fully realised that this was the hindrance for them.

Participant PT2 described how management was supportive of UCD activities by declaring that

> They [management] were making sure we have the time and cycles to do it correctly [...] I believe that management was supportive overall.

Participant PT3 expressed management support to UCD activities by stating that

> They [management] knew I was the expert and I pretty much had free range to do what I needed to do [...] and they trust me to get that done [...]

Participant PT4 discussed management understanding and support to UCD activities by declaring that

> By the time that we started this project [usability team] the company, most people at the company gained good understanding of what are the benefits of making more usable software.

Practice B1.2- Willingness to allocate funds for usability activities and Practice B1.3- Allocating funds for usability activities

The rating for practice B1.2 and B1.3 for CS1, CS2, CS3, CS4, and CS5 was "Yes". Since all case studies exhibited cooperation of management in regards to willingness to allocate funds for conducting usability activities.

Practice B1.4- Presence of dedicated funds for usability activities

The rating for practice B1.4 for CS1, CS2, CS4, and CS5 was "Yes" and "No" for CS3. Although CS3 had funds allocated, yet this fund was dedicated from external bodies and not from the organisation. These funds were used in hiring qualified UCD professionals (CS1, CS2, CS3, CS4, CS5) and

conducting training on UCD related aspects to staff (CS1, CS2, CS3), allocating funds for usability activities (CS1, CS2, CS3, CS4, CS5).

Dimension 3: Usability Practitioners Role

Practice C1.1- Presence of internal usability practitioner

The rating for practice C1.1 for CS1, CS2, CS3, CS4, and CS5 was "Yes". Since all the interviewees were professional usability practitioners whose job roles were: usability product manager, usability analyst, usability engineer, lead user experience designer, and team manager for user experience design respectively.

Practice C1.2- Presence of a usability team led by a usability manager

The rating for practice C1.2 for CS1 and CS4 was "No" and "Yes" for CS2, CS3, and CS5. The value was "No" for CS1 due to the fact that in CS1 there was a usability team of one person, so although there was no usability team led by a usability manager yet there was a usability product manager who was responsible for managing all the user experience aspects of that particular project and designing the user interfaces. So although he was the only UX practitioner yet he had equal authority to the product manager leading to usability aspects gaining high priority in product backlogs.

The value was also "No" for CS4. This was due to the fact that in CS4 there exists a usability team of four people, a lead UX designer running a team of three other designers so in this case there was no usability manager but rather a UX team leader.

Dimension 4: Usability Methods and Techniques

Practice D1.1- Performing usability testing

The rating for practice D1.1 for CS1, CS2, CS3, and CS4 is "Yes" since the design was shown to stakeholders and they were able to perform simulated tasks, whereas it is a "No" for CS5 since usability testing was only conducted on early project phases via low fidelity prototypes.

Participant PT1 performed continual usability testing with high fidelity prototypes with users from a user pool. Participant PT1 described the frequency of usability testing and users' involvement in it by stating that

> At every sort of stage when we had something working we would take it back to one or two of those 6 people [user pool].

Participant PT2 discussed usability testing sessions via RITE method by declaring that

> We do RITE testing so like I typically run two to three users and then through usability testing with high fidelity prototype.

Participant PT3 used remote usability testing for collecting user feedback.

Participant PT4 expressed his desire for users to perform simulated tasks on the software by stating that

> We did not want them to just look at it [software]. I mean looking and using are completely different things.

Participant PT5 discussed the lack of usability testing by declaring that

> For that project all we were able to do was that we did some initial testing upfront.

Participant PT5 discussed the implications of lack of usability testing on the end product by stating that

> So what happened is that at the end of the release we went into this beta phase where they [engineering team] released it to like a limited set of customers and [...] users were unable to purchase completely successfully so they [engineering team] had to pull back the product and spend a couple of months trying to fix it.

Practice D1.2- Planning for usability

The rating for practice D1.2 for CS1, CS2, CS3, CS4, and CS5 is "Yes". In CS1 planning for usability is apparent via various activities: hiring qualified usability product manager, the presence of a user pool, and planning for continual usability testing via high fidelity prototypes. CS2 planned for usability through involving users in ideation phase via brainstorming sessions, conducting contextual inquiry and interviews to identify user needs, and conducting low fidelity prototyping sessions. CS3 and CS4 planned for usability via conducting contextual inquiries and interviews. Moreover, in CS3 users were involved in the design phase via the design studio.

Practice D1.3- Presence of a dedicated usability lab

The rating for practice D1.3 for CS2 and CS5 is "Yes", and "No" for CS1, CS3, and CS4. However, the absence of a dedicated usability lab in CS1 cannot be considered as a shortage from management in providing necessary facilities for usability practitioners. Participant PT1 declared that

management was fully cooperative in regards to the needed tools or facilities. He decided that there is no need for a dedicated usability lab due to reasons expressed in this quote

> I would rather sit down and observe a user in their environment rather than bring them down into a lab setting.

Participant PT2 stated that there exists a number of available usability labs

> We have five dedicated usability labs.

Although participant PT3 did not have any dedicated usability labs yet due to users colocation, this resulted in preference to conduct contextual inquiries, moreover testing was conducted via guerrilla techniques or remotely as a result he felt that there was no need for dedicated usability labs.

Participant PT3 stated his reasons for not needing a dedicated usability lab by declaring that

> In my professional opinion that you can get 80% of what you need with guerrilla tactics and observing users and talking to them and listening to what they are saying or what they are not saying [...] a lot of that stuff [dedicated usability labs] is to me are expensive refinements.

Participant PT4 did not have a dedicated usability lab, yet management was supportive in regards to providing tools needed by usability engineers as expressed by participant PT4

> They [management] provide me with tools as long as it is reasonably priced.

Participant PT5 discussed management support in regards to providing facilities and tools by stating that

> We actually have our own usability lab so that was one thing that was nice so we had an observation room and a testing room back then [...] I think tools were not really an issue, I mean we had all the recording tools and equipment that we needed in order to record the sessions.

Practice D1.4- Utilising a usability reports archive to compile past usability findings

The rating for practice D1.4 for CS1, CS2, CS3, CS4, and CS5 is "Yes". However, participants depended on lightweight methods to document past us-

ability findings.

Participant PT1 described how he utilised Wiki for archiving the results of usability sessions by declaring that

> I keep like notes like on the wiki sort of raw user input [...] sometimes you want to go back because something else comes up and you think that you know particular sessions some user talked about that so you go back to it [Wiki notes].

Participant PT2 discussed how he utilised a patterns library for recording the results of iterations of design solutions by stating that

> It [prototypes] is the recording of them [design solutions] but we also have something we have a pattern library so if I make a decision as a designer that this is the component that we are going to use to handle the situation and I know that this is the component that exist in another places in the application or probably would exist in the future then I create a pattern for it [...] and other designers reference it.

Participant PT3 used Wiki to archive the latest designs.

Participant PT4 used a shared portal for all the design documents and utilised extreme scenario based design. He used a central design record that included documentation and direct mapping between design goals, claims, and usability testing results and the usability engineer documented meeting notes on usability testing sessions.

Participant PT5 used note taking and face to face communication for recording design solution iterations.

Practice D1.5- Performing early user research

The rating for practice D1.5 for CS1, CS2, CS3, CS4, and CS5 is "Yes".

In CS1 early user research was conducted in order to gather user requirements. This involved utilising a user pool and conducting a set of interviews with users to pinpoint their problems and goals. Participant PT1 described the requirement gathering process and the utilisation of user pool in early user research by declaring that

> We picked a few [users] based on product manager recommendations, people [users] that were kind of good at communicating their needs I think we picked also based on the size [of business] so people [users] who did a lot of business and

> people who did little business because the design was you know was dependent on the frequency of use of this particular product so we had a couple of that was really selling like millions on Amazon.com and the others were starting up or were not doing that much, our approach to be actually able to capture both ends of spectrum.

In CS2, CS3, and CS4 early user research was performed in which user requirements were gathered via conducting contextual inquiry and interviews. CS5 performed early user research via studying similar projects then developing low fidelity prototypes and testing them with users.

Participant PT2 discussed his need to conduct contextual inquiry by stating that

> One thing I wanted to do was watch what they [users] do [...] do they [users] have a note book next to them and they [users] diagram it out or do they [users] use MS Visio to diagram it out and [...] and what are all the tiny little details that they [users] need to figure out and how do they [users] need to view them in order to see them all in one place and in order to manage these processes so we [UX team] did a bunch of research upfront before we even started designing.

Participant PT3 described the user requirements gathering process involved in early user research by declaring that

> I would do interviews with stakeholders and end users. I also used contextual inquiry so I would go and I would sit with people [users] occasionally and I would actually do the job myself just to get firsthand experience.

Participant PT5 described the user requirements gathering process involved in early user research and the role of product manager in it by stating that

> The product manager came over with sort of high level requirements. So based on that we actually went on and did some initial validation [...] we initially tested old products and we kind of identified what some of the problem areas were and then we went through and did some initial early concept designs and then we actually tested those with users.

Practice D1.6- Performing iterative design

The rating for practice D1.6 for CS1, CS2, CS3, and CS4 is "Yes" since efforts

were exerted to acquire feedback on design via conducting evaluations at all stages of the development life cycle, and "No" for CS5.

In CS1 design was iterated based on the results of the frequent usability testing sessions and resulted in changes to wire frames or software depending on the feedback and the development phase. Participant PT1 declared changes that the results of user evaluations can lead to by declaring that

> Well often I would go back and modify the wire frames and the user stories just sort of accommodate those results, if they were sort of substantive on some of these actually user sessions would be included, we would have developers sit in and in that case what would happen is that before anything would happen the developers would go and make the changes so you know so that did happen but so it is mixed so if I think there is a little bit of complexity such like just changing a word here or there but if it is a little bit more complex like a new button or a new flow or a new set of fields or something like that then I would go back and change the story and the wire frame.

Participant PT2 discussed changes that occurred as a result of users' feedback throughout the development life cycle by stating that

> Once we get validation that we have a product worth building then we build a very high fidelity prototype [...] and in this test we do, you know, very specific task to gage whether the way we have designed it [software] is usable for them [...] and we do RITE testing so like I typically run two to three users and then through usability testing with high fidelity prototype and then we will take a break for may be half a day or a day and we will regroup as a team and talk about the problems that we [UX team] saw and what we want to try for solutions and the designers will go and iterate on the design and then the next day or the next two days will do another set of users and we just keep basically doing that until we get to the point in which we have resolved all of major problems.

Participant PT4 discussed how the usability engineers had overdone iterative design by iteratively redesigning the software whenever users requested it by stating that

> We were close to the user I would say but one of the problems we had I think in this project is that we did not do a good a job

> as we should have in terms of determining what prioritising changes. I mean the user would want something and we would say OK we will do it, like that without considering the impact, without analysing the request and communicating to users the impact of that request, but OK that is something that we got better at towards the end of the project.

Participant PT5 discussed the lack of iterative design due to lack of design feedback by declaring that

> There was no time to go back in and reverify there was no time to test what was actually implemented you know with the users and there was no opportunity to do that.

Practice D1.7- Quantitative usability metrics can be used to track quality

The rating for practice D1.7 for CS1, CS3, and CS4 is "Yes", and "No" for CS2 and CS5. No reason was given for this by CS2 and CS5.

Practice D1.8- Projects has defined usability goals

The rating for practice D1.8 for CS1, CS2, CS3, and CS4 is "Yes", and "No" for CS5.

In CS1 the project's usability goals are determined via the results of the user requirements gathering, usability testing sessions, and heuristic evaluations. Participant PT1 described the definition of project usability goals by stating that

> We did the sort of usability vision for this project, we had sort of goals based on usability metric [...] we had things like the tasks and what kind of things what the user tasks that we will focus on and what sort of usability matrix would be to measure them. We actually defined those before we started the UI. So which tasks, which flows, so you know we would go by clicks to completion and what exactly we would expect at the end.

In CS3, the project usability goals were derived from the results of heuristic evaluations.

Participant PT3 described how he defined acceptable usability measures by declaring that

> The ergonomics bench marks for like human perception [...] so I look at things loading too fast did the user even notice that anything happened let us slow it down let us put a pause in

> there so I look and I analyse the UI at that kind of micro level from my background.

In CS2 and CS4 usability goals were mainly derived indirectly from users via the results of user requirements gathering and results of usability evaluations.

In CS5 no defined usability goals were set, yet general usability guidelines were used in the development process.

Participant PT4 described how he derived acceptable usability measures from end users by stating that

> [...] Interviews with the clients when we were visioning or creating the vision for the system I mean he [user] would tell us [...] what he wanted, what were the problems with the activity and what they want to accomplish and then based on that we would determine the efficiency matters.

Participant PT5 described how he derived acceptable usability measures via general usability guidelines by declaring that

> I think they are more general guidelines than anything else based on this general usability practices so you know I do not think we had I would not say we had a clear sense matrix I would not say, there was nothing defined by management either so basically going with what we thought good usability practices.

Dimension 5: Strategic Usability

Practice E1.1- Presence of a tracking process for UX quality throughout design projects and across releases

The rating for practice E1.1 for CS2 is "Yes", and "No" for CS1, CS3, CS4, and CS5.

Practice E1.2- Utilising user interface design standards or a centralised definition of preferred design patterns

The rating for practice E1.2 for CS1, CS2, CS3, CS4, and CS5 is "Yes".

Participant PT1 discussed the utilisation of patterns by stating that

> We have both interaction design patterns, we have graphical design patterns, we had a lot of sort of formalization of these different kinds of guidelines.

Participant PT2 described efforts exerted by the UX team towards systematic improvement of usability via the inclusion of useful design solutions into a pattern library by declaring that

> We have a pattern library so if I make a decision as a designer that this is the component that we are going to use to handle the situation and I know that this is the component that exist in other places in the application or probably would exist in the future then I create a pattern [...] and other designers reference it.

Practice E1.3- Projects are prioritised according to the business value of their user experience

The rating for practice E1.3 for CS1, CS3, CS4, and CS5 is "No", and "Unknown" for CS2. The value was "No" for CS5, CS1, CS3, and CS4 since they stated that projects are prioritised according to the business value only and not to the business value of their user experience since it is difficult to identify the return on investment for user experience.

Participant PT4 discussed how projects are prioritised by stating that

> they [projects] are prioritised for their business value period, how much money will they bring.

The conducted interviews attempted to pose questions related to both Nielsen model and UMM-HCS. However, this led participants to become less engaged towards the end of the interview and gave very brief answers to the questions related to practices E1.4, E1.5, E1.6 and E1.7. As a result no significant quotes could be selected for these practices as interviewees resorted to a Yes/ No answer and were reluctant to engage further when they were asked to elaborate their answers.

Practice E1.4- Recognition of the need for user centred design process

The rating for practice E1.4 for CS1, CS2, CS3, CS4, and CS5 is "Yes".

Practice E1.5- Usability data is employed to determine individual projects to be built

The rating for practice E1.5 for CS1, CS2, CS3, CS4, and CS5 is "Yes".

Practice E1.6- Concept of total user experience extend to other forms of customer interaction with the company

The rating for practice E1.6 for CS2 and CS4 is "Yes" and "No" for CS1, CS3, and CS5.

Practice E1.7- User research data is employed to determine overall direction and priorities

The rating for practice E1.7 for CS1 and CS5 is "No" and "Yes" for CS2, CS3, and CS4.

6.2.2 Maturity Level Evaluation of Case Studies Via Nielsen Model

Case Study 1

Table 6.2, column CS1 shows that the closest maturity level that describe CS1 is maturity level 7. However, four practices differed between this case study and the typical case study of maturity level 7. These practices are: practice C1.2, D1.3, E1.1, and E1.3. All four practices had a value of 0 rather than 1 as-is entailed by a typical case study of maturity level 7.

Case Study 2

Table 6.2, column CS2 shows that the closest maturity level that describe CS2 is maturity level 8. However, two practices differed between this case study and the typical case study of maturity level 8. These practices are: practice D1.7 and E1.3. Practice D1.7 had a value of 0 rather than 1 and practice E1.3 had a value of "Do Not Know" rather than 1 as-is entailed by a typical case study of maturity level 8.

Case Study 3

Table 6.2, column CS3 shows that the closest maturity level that describes case study 3 is maturity level 7-8. However, five practices differed between this case study and the typical case study of maturity level 8 and five practices differed between this case study and the typical case study of maturity level 7. The practices that differed from maturity level 7 are practices: B1.4, D1.3, E1.1, E1.3, and E1.7. Practices B1.4, D1.3, E1.1, and E1.3 had a value of 0 rather than 1, whereas practice E1.7 had a value of 1 rather than 0 as-is entailed by a typical case study of maturity level 7.

The practices that differed from maturity level 8 are practices: B1.4, D1.3, E1.1, E1.3, and E1.6. All five practices had a value of 0 rather than 1 as-is entailed by a typical case study of maturity level 8.

Case Study 4

Table 6.2, column CS4 shows that the closest maturity level that describe CS4 is maturity level 8. This is due to the fact that it had four practices that differed between this case study and the typical case study of maturity level 8.

The practices that differed from maturity level 8 are practices: C1.2, D1.3, E1.1, and E1.3. Practices C1.2, D1.3, E1.1, and E1.3 had a value of 0 rather than 1 as-is entailed by a typical case study of maturity level 8.

Case Study 5

Table 6.2, column CS5 shows that CS5 differed significantly from all maturity levels. It differed from maturity level 1, 2, 3, 4, 5, 6, 7, 8 by 15, 14, 14, 11, 8, 8, 9, and 11 practices respectively. Thus it was given a maturity level of "Unknown".

6.3 Revisiting Nielsen Study Aims

The exploratory study that was conducted via Nielsen capability maturity model aimed to investigate the existence of a relationship between the success of AUCDI attempts and usability maturity level and the suitability of Nielsen Corporate Usability Maturity Model for utilisation in assessing usability maturity in the context of Agile projects.

Aim 1: Investigating the existence of a relationship between the success of AUCDI attempts and usability maturity level

This study revealed the existence of a correlation between the success of AUCDI attempts and AUCDI case study's usability maturity level. Successful AUCDI case studies all scored a usability maturity level that ranged from 7-8.

This result can have positive implications on AUCDI practice since practitioners who aim to achieve the integration can utilise Nielsen model before the start of their projects to assess their maturity level. In case of achieving a low maturity level then the different practices included in Nielsen model can be used as a checklist for areas that need to be improved. However, it is important to take into consideration the difference between descriptive and prescriptive maturity models. Descriptive maturity models are used as a diagnostic tool [186] to assess the current capabilities of the examined

entity against specific criteria [16], whereas, prescriptive maturity models aim to pinpoint desirable maturity levels and provide specific and detailed improvement guidelines [16, 186]. Since Nielsen model is a descriptive maturity model, it is limited to acting as a diagnostic tool rather than an improvement tool. Thus the model can assess the performance and pinpoint weak areas but the procedures of improvement of these areas are left to practitioners and are not tackled by the model.

It was not possible to determine the maturity level of CS5, an example of a failed AUCDI attempt as discussed in section 6.2.2. This was due to the wide difference between its achieved practices and any available maturity level. This difficulty of determining the maturity level of CS5 was due to the fact that it had low maturity for all practices related to developers' attitude dimension towards usability: practice A1.1- "Recognition of usability importance", practice A1.2- "Recognition of importance of understanding user needs" and A1.3- "Developers acting on behalf of users". However, it had high maturity for all practices related to the dimensions related to management attitude towards usability, usability practitioners role, usability methods and some practices related to strategic usability. Nevertheless, due to developers' attitude towards usability all the management awareness, allocated funds, usability practitioners and strategic usability practices were not effectively utilised. Moreover, the lack of cooperation of the development team led to discarding a number of practices: practices- D1.1 "Performing usability testing", D1.6- "Performing iterative design", D1.7- "Quantitative usability metrics can be used to track quality", D1.8- "Projects has defined usability goals". The lack of communication among the cross functional team members as a result of geographical separation and engineering team reluctance to collaborate with non engineering teams resulted in the lack of participation of the user experience team and the product manager from participating in the sprint planning and Scrums. This led the engineering team to interpret and implement the designs incorrectly and resulted in lack of iterative refinement of designs.

This implies that the existence of management acknowledgement of usability importance and providing necessary infrastructure for usability activities , i.e., dedicated usability labs, usability professionals, usability funds, usability tools, etc are all less important to project success than developers' attitude towards usability and how they communicate with usability practitioners.

Aim 2: Investigating whether Nielsen Corporate Usability Maturity Model is suitable for utilisation in assessing usability maturity in the context of Agile projects

Nielsen model was not initially developed for Agile software development processes, however, a number of criteria were set in order to investigate the suitability of Nielsen model for utilisation in assessing usability maturity in the context of Agile projects. These criteria involves the following:

CR1: The usability maturity model should not conflict with Agile values and principles

Chapter 2.1, section 2.1.1 and section 2.1.2 discussed the Agile Manifesto and its values and principles. This criteria was set in order to maintain the agility of the development process in case of utilising Nielsen model. However, practice E1.1- "Presence of a tracking process for user experience quality throughout design projects and across releases" could pose a conflict with the Agile value of "Individuals and interactions over processes and tools". This practice also works in support of another Agile principle "Continuous attention to technical excellence and good design enhances agility" since the aim of practice E1.1 is to improve the quality of user experience across all products.

Thus it can be concluded that CR1 is satisfied by Nielsen model since the model does not conflict with Agile values and principles.

CR2: The usability maturity model should integrate UCD activities into the overall project plan and throughout the Agile development life cycle

The reason behind setting this criteria was illustrated in chapter 2, section 2.8 that elaborated that none of the major Agile processes explicitly include guidance for how to develop usable software [178]. In addition, the interaction design role, usability, and UI design in an Agile team is unclear and largely overlooked [20, 45]. Furthermore, principles and practices for understanding and eliciting usability and user requirements and evaluating Agile systems for usability and user experience are generally considerably deficient [156, 178, 267]. In general, it is not yet clear how to incorporate UCD into Agile processes without sacrificing the acknowledged benefits of each of these individual processes.

As a result, criteria CR2 was considered as a significant factor for judging the suitability of Nielsen model for utilisation in assessing usability maturity in the context of Agile projects since the main problem that faces

the Agile domain regarding the integration is when to perform the different UCD activities and how to make them more lightweight in order to accommodate the Agile processes iterative and incremental nature.

Nielsen model includes a variety of UCD activities embodied in the following practices

- D1.1- Performing usability testing.
- D1.2- Planning for usability.
- D1.4- Utilising a usability reports archive to compile past usability findings.
- D1.5- Performing early user research.
- D1.6- Performing iterative design.
- D1.8- Projects has defined usability goals.

However, Nielsen model lacks details on the timing of applying practices D1.1, D1.2, D1.5, and D1.6. Although the model has practice D1.2 that is focused on planning for usability, yet no details were given on what this entails. Practice D1.2 does not clarify what is meant by planning for usability and whether it involves preparation for user research, usability testing or planning to integrate UCD activities into the software development life cycle or all of these issues collectively. Practice D1.4 is focused on utilising a usability reports archive to compile past usability findings, however, the practice does not discuss the form of this usability reports archive to clarify whether it is a lightweight or heavyweight form. Moreover, practice D1.4 focuses only on archiving (documenting) past usability findings without mentioning other aspects that are necessary for AUCDI and that need to be archived: design rationale, source of requirements, results of user research, designs, expected delivery date of designs, etc that are all necessary in the context of Agile projects. Practice D1.5 is focused on performing early user research, however, it uses a generic term to refer to the timing of conducting user research (early), this can imply iteration 0 or sprint 0 in the Agile domain. Moreover, the practice is generic since it does not identify the activities that are involved in performing early user research: identification of user groups, context of use, task analysis, etc. Practice D1.8 is focused on setting defined usability goals for projects yet the problem facing Agile teams is not to set usability goals but rather how to translate these goals into user stories or features in the product back log that can gain priority for execution in the tight Agile time lines and avoid marginalization.

Nielsen model is a generic model, i.e., it is not developed for a particular software development life cycle. As a result the model focus is on declaring the important practices for usability maturity rather than clarifying the timing, method or frequency for conducting these practices along the project plan or the phases of the product development life cycle. This is of specific importance in case of Agile development processes since significant part of the integration challenges are related to the iterative, incremental, tight time line nature of Agile development processes as illustrated in chapter 3, section 3.3.

Thus it can be concluded that criteria CR1 is satisfied by Nielsen model since it does not conflict with Agile values and principle. However, criteria CR2 is not satisfied by Nielsen model since the model does not state clear timings and milestones along the Agile development life cycle for the inclusion of UCD activities into the overall project plan and all phases in the software development life cycle.

6.4 Nielsen Model Critique - A Self Assessment Perspective

This section has a critique to Nielsen model from the perspective of a self assessor who utilised the model on their own to conduct the assessment.

1. **Theoretical Foundations with Respect to Maturation**

   Maturity models should explicate the underpinning theoretical foundations of change and evolution in regards to the investigated class of entities [18, 160]. This involves various issues regarding the method of occurrence of change in the relevant application domain as well as the maturation barriers and drivers as cited in [240].

   The model documentation does not contain any information on the underpinning theoretical foundations of evolution and change with respect to its practices. Moreover no explanation, theoretical or empirical justification was given for the rationale behind choosing practices or placing particular key practices in certain maturity levels except for the personal experience of the author.

   However, Nielsen model declares that as organisations' usability matures they typically progress through the same maturation stages starting by initial hostility and ending by widespread reliance on

user research. It claims that the sequence of maturity stages is fairly universal and consequently any organisation can be matched with this maturity stages description to see what the next stage is likely to be.

2. **Maturity Paths**

   Nielsen model does not provide explicit guidelines that help in either process improvement or upgrading the maturity level. It claims that it takes organisations about twenty years to move from stage 2 (extremely immature usability) to stage 7 (very mature usability), and it takes companies another twenty years to reach the highest maturity level [215].

3. **Empirical Evaluation**

   Hevner et al. [111], stated that researchers need to apply scientific research methods rigourously in designing artefacts, such as maturity models. Thus the elements of maturity models should not be only adapted from previous, normative studies, but should be empirically reasoned [111]. Maturity models should be applied in case studies, incorporated in interviews and surveys, discussed in focus groups, and pretested in pilot groups [57].

   Nielsen model did not disclose any justification for practices except for the personal experience of the author. Thus it can be deduced that the elements of Nielsen maturity model were adapted from prior normative studies rather than being empirically reasoned.

   Moreover, Nielsen's model did not mention any information regarding evaluating the model. There is no reference to any sort of empirical research of using his model. Thus this model is rather abstract, conceptual and based on personal experience and perspective rather than empirically validated research.

4. **Scoring Scheme**

   Although Yes / No as a scoring scheme is considered to be simple and straightforward, however, there is a considerable latitude in each answer for the degree of satisfaction of each specific practice. This latitude needs to be captured or else a lack of information would occur.

5. **Advice on the Assessment of Criteria**

   Assessment methodologies should include a procedural model that guides maturity model users to elaborate on the steps of assessment, their interplay, and the method to elicit the values of the assessed criteria. The assessment results need to be correct, repeatable and accurate ([186], p.25) as cited in [240].

   Nielsen Corporate Usability Maturity Model fails to provide any explicit or detailed guidance for practitioners, there is an absence of the mention of any advice on how to assess the different practices. This is considered an additional burden for a practitioner who is considering to conducting an assessment on his own.

6.5 Nielsen Model Study Challenges and Limitations

Nielsen model is a textual model and thus it had to be carefully examined in order to transform its narrative form into a set of dimensions and practices. These dimensions and practices were used to form a scoring sheet to utilise in conducting the evaluation. This was both time consuming and complicated.

This study suffered from a number of limitations: since a number of AUCDI case studies were purposefully selected in order to reflect a "two tail" design [286] in which cases from both extremes (success and failure) are selected; this endeavour faced publication bias, a phenomena where more positive results are published than negative results [162]. Only one paper in the AUCDI literature [209] reported failure and 46 other papers reported success as it was illustrated in chapter 3, section 3.2.5.3. It would be very beneficial for the research field if more people would report on failure attempts in order to have failure analysis cumulative knowledge so as to come up with counteractive measures.

Moreover, interviews were focused on posing questions regarding published AUCDI case studies in either experience reports or research papers. Although it is preferable not to pose questions regarding events that occurred a long time in the past. Nevertheless, some of these projects were dated back to 2005 and 2008 and this proved to be a bit difficult for both the researcher and participants. As for the participants they had to exert

an effort in remembering some of the details. The researcher had to be alert to redirect the participants to answer the question about the time of this past project rather than the current time.

The conducted interviews attempted to pose questions related to both Nielsen model and UMM-HCS. This made participants less engaged towards the end of the interview and as a result they gave very brief answers to the questions related to practices E1.4, E1.5, E1.6 and E1.7. Thus no significant quotes could be selected for these practices as interviewees resorted to a Yes/ No answer and were reluctant to engage further when they were asked to elaborate their answers.

6.6 Conclusion

The novelty of this work lies in two issues: first, carrying out an empirical work that has not been approached before and utilising a specific technique and utilising it to a novel area [237]; this was achieved via transforming Nielsen model from textual form into a set of measurable dimensions and practices and applying it to five AUCDI case studies in order to evaluate their usability maturity levels. These case studies allowed for the empirical validation of the model. Moreover, this empirical study resulted in a critique to the model that can be used as a foundation for recommendations for both designing UMMs and also an Agile UCD maturity model as it will be illustrated in chapter 9 that proposes a maturity model for integrating Agile development processes and UCD.

The second reason for the novelty of this work is approaching non tackled areas, resulting in an addition to knowledge [237], since this research looked into a new area; which is the existence of a relationship between usability maturity levels and the success of AUCDI and also the suitability of current UMMs for application on AUCDI case studies.

The investigation of the existence of a relationship between the success of AUCDI attempts and usability maturity level revealed the existence of a correlation between the success of AUCDI attempts and the AUCDI case study's usability maturity level since successful AUCDI case studies all scored a usability maturity level that ranged from 7-8. In the failed AUCDI attempt, it was not possible to determine its maturity level as discussed in section 6.2.2 and as a result it was given a maturity level of "Unknown".

Moreover, the investigation of the suitability of UMMs for utilisation in

assessing usability maturity in the context of Agile projects gave an indication that the model does not conflict with Agile values and principles. However, although the model has a variety of UCD activities, yet, it lacks details on the timing of applying the different practices along the Agile development life cycle iterations or sprints. This is of specific importance in case of Agile development processes as significant part of the integration challenges that faces the Agile domain, as illustrated in chapter 3, section 3.3 and chapter 6, section 6.3 is related to the timing of performing the different UCD activities in order to accommodate the Agile processes iterative and incremental nature.

Although the existence of a correlation between the success of AUCDI attempts and the AUCDI case study's usability maturity level result can have positive implications on AUCDI practice since practitioners who aim to achieve the integration can utilise Nielsen model before the start of their projects to assess their maturity level. In case of achieving a low maturity level then the different practices included in Nielsen model can be used as a checklist for areas that need to be improved. However, two issues are of concern: first, Nielsen model is a descriptive maturity model, thus it is limited to acting as a diagnostic tool rather than an improvement tool, so the model can assess the performance and pinpoint weak areas but the procedures of improvement of these areas are left to practitioners and are not tackled by the model. Second, the lack of timing for the different UCD activities can hinder the usefulness of the model since organisations can use it as a checklist of the UCD activities that need to be performed, however, when they initiate their software development process they will be more concerned with the timing and the lightweight method through which they can achieve the recommended UCD activities by Nielsen model.

Moreover, an open issue is the further AUCDI challenges that are specific to the Agile domain and that are not tackled in Nielsen model. Those AUCDI challenges were discussed in detail in chapter 3, section 3.3. These issues need to be taken into consideration by any researcher who considers developing a UMM in the context of Agile projects. AUCDI challenges that are not approached by Nielsen model are: practices regarding the communication, coordination and collaboration between UCD practitioners and Agile developers in order to synchronize and complete their work, practices related to design modularization and chunking, UCD practitioner workload, and maintaining communication between the customer and the development team. Another issue that needs to be approached by the developers of UMMs for Agile development processes is the features

and activities that should be played by some team roles. These team roles are XP coach, Scrum master, and product manager whose role can impact the integration process.

Though this exploratory study has helped to elucidate the phenomena being studied by suggesting that there exists a correlation between the success of AUCDI attempts and the AUCDI case studies' usability maturity levels. Yet, several reasons suggested that further investigation is required. First, the critique that was raised to Nielsen model. Second, the inability of Nielsen model to assess the maturity level of the failed AUCDI case study. Third, the presence of UMM-HCS model as a suitable model for assessment as shown in table 5.2 since it is lightweight, generic, and empirically evaluated. Finally, the desire to conduct a comparative study between the results of applying Nielsen model and UMM-HCS on AUCDI case studies. As a result, UMM-HCS model was utilised on the same five AUCDI case studies and the results are reported in chapter 7.

6.7 Chapter Summary

This chapter investigated the suitability of Nielsen Corporate Usability Maturity Model for utilisation in the Agile domain in order to assess the organisation's UCD capability. It reported on applying Nielsen Model in five case studies that integrated Agile development processes and user centred design and utilising the model in assessing their usability maturity level. It offered a critique to Nielsen Corporate Usability Maturity Model from a self assessment perspective. The chapter ended by listing the research challenges faced and the research limitations.

The following chapter will report on applying UMM-HCS model in the same five case studies that integrated Agile development processes and UCD and utilising the model in assessing their usability maturity level.

Chapter 7

An Empirical Study of Usability Maturity Model-Human Centredness Scale

This chapter is a continuation of the investigation that started in chapter 6, of the suitability of UMMs for utilisation in the context of Agile projects. It reports on applying Usability Maturity Model-Human Centredness Scale (UMM-HCS) in five case studies that integrated Agile development processes and UCD and utilising the model in assessing their usability maturity level. An analysis and general observations are provided to the suitability of UMM-HCS in the context of Agile projects. The chapter offers a critique to UMM-HCS from a self assessment perspective. The chapter also provides a comparison and mapping of Nielsen Model and UMM-HCS Model. The chapter ends by listing the research challenges and research limitations.

7.1 Utilising UMM-HCS in AUCDI Case Studies

The European INUSE project developed UMM-HCS [68] in 1998. This model is derived from all UMMs available until the middle phase of the INUSE project. These models were: the Total Systems Maturity (TSM) model [155], the ULMM [87], UCDM [70] and Crosby [49] as shown in chapter 5, figure 5.1.

Table 7.1 shows how UMM-HCS embodies usability maturity via 6 maturity levels. These maturity levels are: Level X-Unrecognised, Level A-Recognised, Level B-Considered, Level C-Implemented, Level D-Integrated, and Level E-Institutionalized. These maturity levels are defined by a set of attributes, these attributes are embodied in a set of attitude, technology and / or management activities that are performed at that level as shown in table 7.1.

ID	**Title**
Level X	**Unrecognised**
	(no indicators)
Level A	**Recognised**
A1	Problem recognition attribute
A2	Performed processes attribute
Level B	**Considered**
B.1	Quality in use awareness attribute
B.2	User focus attribute
Level C	**Implemented**
C.1	User involvement attribute
C.2	Human factors technology attribute
C.3	Human factors skills attribute
Level D	**Integrated**
D.1	Integration attribute
D.2	Improvement attribute
D.3	Iteration attribute
Level E	**Institutionalized**
E.1	Human-centred leadership attribute
E.2	Organisational human-centredness attribute

Table 7.1: UMM-HCS Maturity Levels and Process Attributes [68]

These practices are used in assessing the user centred approach of an organisation in regards to its working culture and systems development and

support activities. This assessment occurs via a scoring scheme that is shown in table 7.2. The majority of UMM-HCS attributes are management practices yet some attributes are focused on measuring management or staff attitudes or technical capabilities [68]. UMM-HCS offers organisations an understanding of how organisation's usability maturity progress and allows organisations to measure their maturity and subsequently plan for improvement or utilise the model in organisational design so as to ensure a usability focused approach.

Abbreviation	**Rating**	**Description**
N	Not Achieved	There is no evidence of achievement of the defined practice
P	Partially Achieved	There is some achievement of the defined practice
L	Largely Achieved	There is significant achievement of the defined practice
F	Fully Achieved	There is full achievement of the defined practice

Table 7.2: UMM-HCS Scoring Scheme [68]

Table 7.3 shows the rating of realisation of each attribute in order to achieve a particular maturity level [68]. UMM-HCS has an assessment recording form, that is shown in table 7.4, and is used by assessors to record the ratings of the different attributes. Its documentation offers a detailed description to the use of the recording form [68]. UMM-HCS can be used either as a stand alone model for assessing usability, or as a companion to ISO 15504-compliant process reference or any other process assessment model that does not have UCD as its focus [68]. Where the column 1, 2, 3 headings signify the first, second and third interviews since it is advised to hold interviews with the systems development group head and a project manager. However, since most of the evaluated practices are related to usability and user centred design issues thus it was preferred to interview lead usability practitioners in order to ensure interviewer's ability to answer questions related to the evaluated practices. UMM-HCS author was contacted for an updated version of the documentation and was contacted on several occasions via emails in order to clarify some issues related to UMM-HCS model practices, terminology, assessment criteria and assessment method.

Answers to interview questions were used in evaluating the maturity level of each case study and results are reported in tables 7.5, 7.6, 7.7, 7.8, and

Scale	Process Attributes	Rating
Level A	Problem recognition	Fully or largely
	Performed processes	Fully or largely
Level B	Problem recognition	Fully
	Performed processes	Fully
	Quality in use awareness	Fully or largely
	User focus	Fully or largely
Level C	Problem recognition	Fully
	Performed processes	Fully
	Quality in use awareness	Fully
	User focus	Fully
	User involvement	Fully or largely
	HF technology	Fully or largely
	HF skills	Fully or largely
Level D	Problem recognition	Fully
	Performed processes	Fully
	Quality in use awareness	Fully
	User focus	Fully
	User involvement	Fully
	HF technology	Fully
	HF skills	Fully
	Integration	Fully or largely
	Improvement	Fully or largely
	Iteration	Fully or largely
Level E	Problem recognition	Fully
	Performed processes	Fully
	Quality in use awareness	Fully
	User focus	Fully
	User involvement	Fully
	HF technology	Fully
	HF skills	Fully
	Integration	Fully
	Improvement	Fully
	Iteration	Fully
	Human-centred leadership	Fully or largely
	Organisational Human-centredness	Fully or largely

Table 7.3: UMM-HCS Maturity Level Ratings [68]

7.9.

Table 7.4 shows that each maturity level is composed of one or more attribute. UMM-HCS documentation [68] contains further detail on each of the attributes and on the use of the recording form in order to record the rating of the different attributes.

UMM-HCS documentation provides guidelines on rating the attributes. These guidelines involve rating the different practices assessed and asking for evidence. It states that the benefit of doubt should be given in assessing the achievement of a particular practice and accordingly round up the rating to the higher level of achievement. The rating of each attribute should be combined via utilising the benefit of doubt and rounding up if required. In case of conducting more than one interview, the box at the rightmost column shown in table 7.4 should include the combination of ratings. The rating of all attributes at each level should be combined in the box in the bottom right hand corner of each table. These guidelines were followed in rating the five AUCDI case studies as it will be illustrated in tables 7.5, 7.6, 7.7, 7.8, and 7.9.

Level A Recognised					
A.1	**Problem Recognition Attribute**	**1**	**2**	**3**	**Rating**
A1.1	Problem Recognition				
	Combined Rating for Attribute (A1.1):				
A.2	**Performed Processes Attribute**				
A2.1	Information collection				
A2.2	Performance of relevant practices				
	Combined Rating for Attribute (A2.1 to 2.2):				
	Combination of Ratings for this Level:				
Level B Considered					
B.1	**Quality in Use Awareness Attribute**	**1**	**2**	**3**	**Rating**
B1.1	Quality in use training				
B1.2	Human-centred methods training				
B1.3	Human-system interaction training				
	Combined Rating for Attribute (B1.1 and 1.3):				
B.2	**User Focus Attribute**				
B2.1	User consideration training				
B2.2	Context of use training				
	Combined Rating for Attribute (B2.1 and 2.2):				
	Combination of Ratings for Level:				
Level C Implemented					
C.1	**User Involvement Attribute**	1	2	3	Rating
C1.1	Active involvement of users				
C1.2	Elicitation of user experience				
C1.3	End users define quality-in-use				
C1.4	Continuous evaluation				
	Combined Rating for Attribute (C1.1 to 1.4):				
C.2	**HF Technology Attribute**				
C2.1	Provide appropriate human-centred methods				
C2.2	Provide suitable facilities and tools				
C2.3	Maintain quality in use techniques				
	Combined Rating for Attribute (C2.1 to 2.3):				
C.3	**HF Skills Attribute**				
C3.1	Decide on required skills				
C3.2	Develop appropriate skills				
C3.3	Deploy appropriate staff				
	Combined Rating for Attribute (C3.1 to 3.3):				
	Combination of Ratings for this Level:				
Level D Integrated					
D.1	**Integration Attribute**	**1**	**2**	**3**	**Rating**
D1.1	Integrate HF processes				
D1.2	Facilitate interface between HF and the organisation				
D1.3	Use appropriate representations				
	Combined Rating for Attribute (D1.1 to 1.3):				
D.2	**Improvement Attribute**				
D2.1	Ensure design feedback				
D2.2	Change based on feedback				
D2.3	Timing of feedback				
	Combined Rating for Attribute (D2.1 to 2.3):				
D.3	**Iteration Attribute**				
D3.1	Minimize risks by iteration of design				
D3.2	Manage iteration of design solutions				
D3.3	Use design objectives to control iteration				
	Combined Rating for Attribute (D3.1 to 3.3):				
	Combination of Ratings for this Level:				
Level E Institutionalized					
E.1	**Human-Centred Leadership Attribute**	**1**	**2**	**3**	**Rating**
E1.1	Manage usability programme				
E1.2	Systematic improvement in quality in use				
E1.3	Human-centred improvement of organisation				
	Combined Rating for Attribute (E1.1 to 1.3):				
E.2	**Organisational Human-Centredness Attribute**				
E2.1	Organisational implementation of user-centred practices				
E2.2	Acceptance of human-centred skills				
	Combined Rating for Attribute (E2.1 to 2.2):				
	Combination of Ratings for this Level:				

Table 7.4: UMM-HCS Recording Form [68]

7.2 Results

This section reports on applying UMM-HCS in five AUCDI case studies that were discussed in chapter 5 and utilising UMM-HCS in assessing their usability maturity level as reported in tables 7.5, 7.6, 7.7, 7.8, and 7.9.

Level A Recognised			
A.1	**Problem Recognition Attribute**	**1**	**Rating**
A1.1	Problem Recognition	Fully	
	Combined Rating for Attribute (A1.1):	**Fully**	
A.2	**Performed Processes Attribute**		
A2.1	Information collection	Fully	
A2.2	Performance of relevant practices	Fully	
	Combined Rating for Attribute (A2.1 to 2.2):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level B Considered			
B.1	**Quality in Use Awareness Attribute**	**1**	**Rating**
B1.1	Quality in use training	Fully	
B1.2	Human-centred methods training	Fully	
B1.3	Human-system interaction training	Fully	
	Combined Rating for Attribute (B1.1 and 1.3):	**Fully**	
B.2	**User Focus Attribute**		
B2.1	User consideration training	Fully	
B2.2	Context of use training	Fully	
	Combined Rating for Attribute (B2.1 and 2.2):	**Fully**	
	Combination of Ratings for Level:		**Fully**
Level C Implemented			
C.1	**User Involvement Attribute**	**1**	**Rating**
C1.1	Active involvement of users	Fully	
C1.2	Elicitation of user experience	Fully	
C1.3	End users define quality-in-use	Largely	
C1.4	Continuous evaluation	Fully	
	Combined Rating for Attribute (C1.1 to 1.4):	**Fully**	
C.2	**HF Technology Attribute**		
C2.1	Provide appropriate human-centred methods	Fully	
C2.2	Provide suitable facilities and tools	Fully	
C2.3	Maintain quality in use techniques	Largely	
	Combined Rating for Attribute (C2.1 to 2.3):	**Fully**	
C.3	**HF Skills Attribute**		
C3.1	Decide on required skills	Partially	
C3.2	Develop appropriate skills	Largely	
C3.3	Deploy appropriate staff	Largely	
	Combined Rating for Attribute (C3.1 to 3.3):	**Largely**	
	Combination of Ratings for this Level:		**Fully**
Level D Integrated			
D.1	**Integration Attribute**	**1**	**Rating**
D1.1	Integrate HF processes	Partially	
D1.2	Facilitate interface between HF and the organisation	Largely	
D1.3	Use appropriate representations	Largely	
	Combined Rating for Attribute (D1.1 to 1.3):	**Largely**	
D.2	**Improvement Attribute**		
D2.1	Ensure design feedback	Fully	
D2.2	Change based on feedback	Largely	
D2.3	Timing of feedback	Fully	
	Combined Rating for Attribute(D2.1 to 2.3):	**Fully**	
D.3	**Iteration Attribute**		
D3.1	Minimize risks by iteration of design	Largely	
D3.2	Manage iteration of design solutions	Partially	
D3.3	Use design objectives to control iteration	Largely	
	Combined Rating for Attribute (D3.1 to 3.3):	**Largely**	
	Combination of Ratings for this Level:		**Largely**
Level E Institutionalized			
E.1	**Human-Centred Leadership Attribute**	**1**	**Rating**
E1.1	Manage usability programme	Not achieved	
E1.2	Systematic improvement in quality in use	Not achieved	
E1.3	Human-centred improvement of organisation	Not achieved	
	Combined Rating for Attribute (E1.1 to 1.3):	**Not achieved**	
E.2	**Organisational Human-Centredness Attribute**		
E2.1	Organisational implementation of user-centred practices	Not achieved	
E2.2	Acceptance of human-centred skills	Partially	
	Combined Rating for Attribute (E2.1 to 2.2):	**Partially**	
	Combination of Ratings for this Level:		**Partially**

Table 7.5: UMM-HCS Recording Form-Case Study 1

Level A Recognised			
A.1	**Problem Recognition Attribute**	**1**	**Rating**
A1.1	Problem Recognition	Fully	
	Combined Rating for Attribute (A1.1):	**Fully**	
A.2	**Performed Processes Attribute**		
A2.1	Information collection	Fully	
A2.2	Performance of relevant practices	Fully	
	Combined Rating for Attribute (A2.1 to 2.2):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level B Considered			
B.1	**Quality in Use Awareness Attribute**	**1**	**Rating**
B1.1	Quality in use training	Partially	
B1.2	Human-centred methods training	Not achieved	
B1.3	Human-system interaction training	Not achieved	
	Combined Rating for Attribute (B1.1 and 1.3):	**Not achieved**	
B.2	**User Focus Attribute**		
B2.1	User consideration training	Not achieved	
B2.2	Context of use training	Not achieved	
	Combined Rating for Attribute (B2.1 and 2.2):	**Not achieved**	
	Combination of Ratings for Level:		**Not achieved**
Level C Implemented			
C.1	**User Involvement Attribute**	**1**	**Rating**
C1.1	Active involvement of users	Fully	
C1.2	Elicitation of user experience	Fully	
C1.3	End users define quality-in-use	Fully	
C1.4	Continuous evaluation	Fully	
	Combined Rating for Attribute (C1.1 to 1.4):	**Fully**	
C.2	**HF Technology Attribute**		
C2.1	Provide appropriate human-centred methods	Fully	
C2.2	Provide suitable facilities and tools	Fully	
C2.3	Maintain quality in use techniques	Fully	
	Combined Rating for Attribute (C2.1 to 2.3):	**Fully**	
C.3	**HF Skills Attribute**		
C3.1	Decide on required skills	Fully	
C3.2	Develop appropriate skills	Fully	
C3.3	Deploy appropriate staff	Largely	
	Combined Rating for Attribute (C3.1 to 3.3):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level D Integrated			
D.1	**Integration Attribute**	**1**	**Rating**
D1.1	Integrate HF processes	Largely	
D1.2	Facilitate interface between HF and the organisation	Fully	
	Combined Rating for Attribute (D1.1 to 1.3):	**Fully**	
D.2	**Improvement Attribute**		
D2.1	Ensure design feedback	Fully	
D2.2	Change based on feedback	Fully	
D2.3	Timing of feedback	Fully	
	Combined Rating for Attribute(D2.1 to 2.3):	**Fully**	
D.3	**Iteration Attribute**		
D3.1	Minimize risks by iteration of design	Fully	
D3.2	Manage iteration of design solutions	Fully	
D3.3	Use design objectives to control iteration	Fully	
	Combined Rating for Attribute (D3.1 to 3.3):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level E Institutionalized			
E.1	**Human-Centred Leadership Attribute**	**1**	**Rating**
E1.1	Manage usability programme	Fully	
E1.2	Systematic improvement in quality in use	Fully	
E1.3	Human-centred improvement of organisation	Fully	
	Combined Rating for Attribute (E1.1 to 1.3):	**Fully**	
E.2	**Organisational Human-Centredness Attribute**		
E2.1	Organisational implementation of user-centred practices	Fully	
E2.2	Acceptance of human-centred skills	Fully	
	Combined Rating for Attribute (E2.1 to 2.2):	**Fully**	
	Combination of Ratings for this Level:		**Fully**

Table 7.6: UMM-HCS Recording Form-Case Study 2

Level A Recognised			
A.1	**Problem Recognition Attribute**	**1**	**Rating**
A1.1	Problem Recognition	Fully	
	Combined Rating for Attribute (A1.1):	**Fully**	
A.2	**Performed Processes Attribute**		
A2.1	Information collection	Fully	
A2.2	Performance of relevant practices	Fully	
	Combined Rating for Attribute (A2.1 to 2.2):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level B Considered			
B.1	**Quality in Use Awareness Attribute**	**1**	**Rating**
B1.1	Quality in use training	Fully	
B1.2	Human-centred methods training	Fully	
B1.3	Human-system interaction training	Not achieved	
	Combined Rating for Attribute (B1.1 and 1.3):	**Fully**	
B.2	**User Focus Attribute**		
B2.1	User consideration training	Largely	
B2.2	Context of use training	Largely	
	Combined Rating for Attribute (B2.1 and 2.2):	**Largely**	
	Combination of Ratings for Level:		**Fully**
Level C Implemented			
C.1	**User Involvement Attribute**	**1**	**Rating**
C1.1	Active involvement of users	Fully	
C1.2	Elicitation of user experience	Fully	
C1.3	End users define quality-in-use	Largely	
C1.4	Continuous evaluation	Fully	
	Combined Rating for Attribute (C1.1 to 1.4):	**Fully**	
C.2	**HF Technology Attribute**		
C2.1	Provide appropriate human-centred methods	Fully	
C2.2	Provide suitable facilities and tools	Fully	
C2.3	Maintain quality in use techniques	Largely	
	Combined Rating for Attribute (C2.1 to 2.3):	**Fully**	
C.3	**HF Skills Attribute**		
C3.1	Decide on required skills	Largely	
C3.2	Develop appropriate skills	Partially	
C3.3	Deploy appropriate staff	Largely	
	Combined Rating for Attribute (C3.1 to 3.3):	**Largely**	
	Combination of Ratings for this Level:		**Fully**
Level D Integrated			
D.1	**Integration Attribute**	**1**	**Rating**
D1.1	Integrate HF processes	Partially	
D1.2	Facilitate interface between HF and the organisation	Largely	
D1.3	Use appropriate representations	Largely	
	Combined Rating for Attribute (D1.1 to 1.3):	**Largely**	
D.2	**Improvement Attribute**		
D2.1	Ensure design feedback	Fully	
D2.2	Change based on feedback	Largely	
D2.3	Timing of feedback	Fully	
	Combined Rating for Attribute (D2.1 to 2.3):	**Fully**	
D.3	**Iteration Attribute**		
D3.1	Minimize risks by iteration of design	Largely	
D3.2	Manage iteration of design solutions	Partially	
D3.3	Use design objectives to control iteration	Largely	
	Combined Rating for Attribute (D3.1 to 3.3):	**Largely**	
	Combination of Ratings for this Level:		**Largely**
Level E Institutionalized			
E.1	**Human-Centred Leadership Attribute**	**1**	**Rating**
E1.1	Manage usability programme	Partially	
E1.2	Systematic improvement in quality in use	Partially	
E1.3	Human-centred improvement of organisation	Partially	
	Combined Rating for Attribute (E1.1 to 1.3):	**Partially**	
E.2	**Organisational Human-Centredness Attribute**		
E2.1	Organisational implementation of user-centred practices	Largely	
E2.2	Acceptance of human-centred skills	Largely	
	Combined Rating for Attribute (E2.1 to 2.2):	**Largely**	
	Combination of Ratings for this Level:		**Largely**

Table 7.7: UMM-HCS Recording Form-Case Study 3

Level A Recognised			
A.1	**Problem Recognition Attribute**	**1**	**Rating**
A1.1	Problem Recognition	Fully	
	Combined Rating for Attribute (A1.1):	**Fully**	
A.2	**Performed Processes Attribute**		
A2.1	Information collection	Fully	
A2.2	Performance of relevant practices	Fully	
	Combined Rating for Attribute (A2.1 to 2.2):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level B Considered			
B.1	**Quality in Use Awareness Attribute**	**1**	**Rating**
B1.1	Quality in use training	Partially	
B1.2	Human-centred methods training	Partially	
B1.3	Human-system interaction training	Not achieved	
	Combined Rating for Attribute (B1.1 and 1.3):	**Partially**	
B.2	**User Focus Attribute**		
B2.1	User consideration training	Partially	
B2.2	Context of use training	Partially	
	Combined Rating for Attribute (B2.1 and 2.2):	**Partially**	
	Combination of Ratings for Level:		**Partially**
Level C Implemented			
C.1	**User Involvement Attribute**	**1**	**Rating**
C1.1	Active involvement of users	Largely	
C1.2	Elicitation of user experience	Fully	
C1.3	End users define quality-in-use	Fully	
C1.4	Continuous evaluation	Fully	
	Combined Rating for Attribute (C1.1 to 1.4):	**Fully**	
C.2	**HF Technology Attribute**		
C2.1	Provide appropriate human-centred methods	Largely	
C2.2	Provide suitable facilities and tools	Largely	
C2.3	Maintain quality in use techniques	Largely	
	Combined Rating for Attribute (C2.1 to 2.3):	**Largely**	
C.3	**HF Skills Attribute**		
C3.1	Decide on required skills	Largely	
C3.2	Develop appropriate skills	Partially	
C3.3	Deploy appropriate staff	Largely	
	Combined Rating for Attribute (C3.1 to 3.3):	**Largely**	
	Combination of Ratings for this Level:		**Largely**
Level D Integrated			
D.1	**Integration Attribute**	**1**	**Rating**
D1.1	Integrate HF processes	Not achieved	
D1.2	Facilitate interface between HF and the organisation	Partially	
D1.3	Use appropriate representations	Partially	
	Combined Rating for Attribute (D1.1 to 1.3):	**Partially**	
D.2	**Improvement Attribute**		
D2.1	Ensure design feedback	Fully	
D2.2	Change based on feedback	Fully	
D2.3	Timing of feedback	Fully	
	Combined Rating for Attribute(D2.1 to 2.3):	**Fully**	
D.3	**Iteration Attribute**		
D3.1	Minimize risks by iteration of design	Fully	
D3.2	Manage iteration of design solutions	Fully	
D3.3	Use design objectives to control iteration	Largely	
	Combined Rating for Attribute (D3.1 to 3.3):	**Fully**	
	Combination of Ratings for this Level:		**Fully**
Level E Institutionalized			
E.1	**Human-Centred Leadership Attribute**	**1**	**Rating**
E1.1	Manage usability programme	Partially	
E1.2	Systematic improvement in quality in use	Not achieved	
E1.3	Human-centred improvement of organisation	Not achieved	
	Combined Rating for Attribute (E1.1 to 1.3):	**Not achieved**	
E.2	**Organisational Human-Centredness Attribute**		
E2.1	Organisational implementation of user-centred practices	Partially	
E2.2	Acceptance of human-centred skills	Partially	
	Combined Rating for Attribute (E2.1 to 2.2):	**Partially**	
	Combination of Ratings for this Level:		**Partially**

Table 7.8: UMM-HCS Recording Form-Case Study 4

Level A Recognised			
A.1	**Problem Recognition Attribute**	**1**	**Rating**
A1.1	Problem Recognition	Partially	
	Combined Rating for Attribute (A1.1):	**Partially**	
A.2	**Performed Processes Attribute**		
A2.1	Information collection	Fully	
A2.2	Performance of relevant practices	Partially	
	Combined Rating for Attribute (A2.1 to 2.2):	**Unknown**	
	Combination of Ratings for this Level:		**Unknown**
Level B Considered			
B.1	**Quality in Use Awareness Attribute**	**1**	**Rating**
B1.1	Quality in use training	Partially	
B1.2	Human-centred methods training	Partially	
B1.3	Human-system interaction training	Not achieved	
	Combined Rating for Attribute (B1.1 and 1.3):	**Partially**	
B.2	**User Focus Attribute**		
B2.1	User consideration training	Partially	
B2.2	Context of use training	Partially	
	Combined Rating for Attribute (B2.1 and 2.2):	**Partially**	
	Combination of Ratings for Level:		**Partially**
Level C Implemented			
C.1	**User Involvement Attribute**	**1**	**Rating**
C1.1	Active involvement of users	Partially	
C1.2	Elicitation of user experience	Partially	
C1.3	End users define quality-in-use	No	
C1.4	Continuous evaluation	Partially	
	Combined Rating for Attribute (C 1.1 to 1.4):	**Partially**	
C.2	**HF Technology Attribute**		
C2.1	Provide appropriate human-centred methods	Partially	
C2.2	Provide suitable facilities and tools	Fully	
C2.3	Maintain quality in use techniques	Partially	
	Combined Rating for Attribute (C2.1 to 2.3):	Partially	
C.3	**HF Skills Attribute**		
C3.1	Decide on required skills	Not achieved	
C3.2	Develop appropriate skills	Partially	
C3.3	Deploy appropriate staff	Partially	
	Combined Rating for Attribute (C3.1 to 3.3):	**Partially**	
	Combination of Ratings for this Level:		**Partially**
Level D Integrated			
D.1	**Integration Attribute**	**1**	**Rating**
D1.1	Integrate HF processes	Not achieved	
D1.2	Facilitate interface between HF and the organisation	Not achieved	
D1.3	Use appropriate representations	Partially	
	Combined Rating for Attribute (D1.1 to 1.3):	**Not achieved**	
D.2	**Improvement Attribute**		
D2.1	Ensure design feedback	Partially	
D2.2	Change based on feedback	Not achieved	
D2.3	Timing of feedback	Not achieved	
	Combined Rating for Attribute (D2.1 to 2.3):	**Not achieved**	
D.3	**Iteration Attribute**		
D3.1	Minimize risks by iteration of design	Partially	
D3.2	Manage iteration of design solutions	Not achieved	
D3.3	Use design objectives to control iteration	Not achieved	
	Combined Rating for Attribute (D3.1 to 3.3):	**Not achieved**	
	Combination of Ratings for this Level:		**Not achieved**
Level E Institutionalized			
E.1	**Human-Centred Leadership Attribute**	**1**	**Rating**
E1.1	Manage usability programme	Not achieved	
E1.2	Systematic improvement in quality in use	Not achieved	
E1.3	Human-centred improvement of organisation	Not achieved	
	Combined Rating for Attribute (E1.1 to 1.3):	**Not achieved**	
E.2	**Organisational Human-Centredness Attribute**		
E2.1	Organisational implementation of user-centred practices	Not achieved	
E2.2	Acceptance of human-centred skills	Partially	
	Combined Rating for Attribute (E2.1 to 2.2):	**Partially**	
	Combination of Ratings for this Level:		**Partially**

Table 7.9: UMM-HCS Recording Form-Case Study 5

7.2.1 Rating of UMM-HCS Process Attributes

This section discusses the findings from applying UMM-HCS in five AUCDI case studies. Appendix I has detailed description of the practices that typify each maturity level [68].

A.1- Problem Recognition Attribute

Problem recognition attribute is composed of one practice: A1.1- Problem recognition.

The rating for Problem recognition attribute for CS1, CS2, CS3, and CS4 is "Fully" since management support to usability aspects was evident via a number of issues. These issues were: allocating funds for usability activities (CS1, CS2, CS3, CS4, CS5), hiring qualified UCD professionals (CS1, CS2, CS3, CS4, CS5), conducting training on UCD related aspects to staff (CS1, CS2, CS3), close customer collaboration (CS4), close user collaboration (CS1, CS2, CS3, CS4). In addition for CS1, a lead UCD professional (usability product owner) was hired and offered a a high organisational position that is equivalent to that of the product manager in order to ensure that usability issues will not take less priority than functional issues. In CS2 a UX team was set as a centralised group that is assigned to the different projects. In CS5 it was "Partially" since although management was supportive to usability aspects via dedicating funds for usability activities and hiring qualified UCD professionals, yet staff was not supportive at all as it will be illustrated by participant PT5 quote.

Participant PT3 expressed management support to UCD activities by stating that

> As far as specific support for the design studio itself from a management perspective it was kind of non issue [...] it would be if [...] [Participant PT2 name] wants to do this go ahead.

Participant PT4 discussed several factors that led to management support to UCD activities by stating that

> They [management] were getting more and more feedback from the customers that they needed to make their software more usable and they were also getting more pressure from their competitors from their company competitors who were trying to gain competitive advantage through more usable software.

Staff were also supportive to usability aspects in CS1, CS2, and CS3 since

it resulted in producing a better product (CS1 and CS2) and taking the burden of usability away from them (CS1). In CS4, staff were not initially supportive yet after participant PT4 illustrated the benefits of UCD they started to be more appreciative to his work.

Participant PT1 discussed the reasons behind the supportive attitude of developers to new usability related team roles by stating that

> It [having a usability product owner] was much easier they [developers] did not have to do things over and over again because it felt better when people would say this works really well.

Participant PT2 discussed developers motives for supporting usability by stating that

> They [developers] did not want to have to build this twice [...] they definitely wanted us to prototype and test it and get it right before they jumped in and started developing.

Participant PT3 discussed the developers cooperation in attending training related to different UCD aspects by stating that

> [...] out of 8 developers we had four participate in the first design studio [training] [...] so after the first one [design studio training] the feedback from their peers [developers' peers] brought the rest of them [developers] in.

Participant PT4 described his efforts to educate staff on usability engineering and how staff appreciation for usability engineering increased gradually by stating that

> I would work with individual teams first and show them and do usability designs and run usability evaluations to get the feedback for the design, show them the benefits of what I was doing and that kind of bubbled up to more senior people.

Participant PT5 described the lack of understanding and support of the development team to usability activities by stating that

> They [the engineering team] were working on their own based on their own schedule basically working in sort of a silo and moving that way so I think from the engineering stand point I do not think they really understood the value of user experience.

A.2- Performed Process Attribute

Performed process attribute is composed of two practices: practice A2.1- Information collection and practice A2.2- Performance of relevant practices.

Practice A2.1- Information Collection

The rating for practice A2.1 for CS1, CS2, CS3, CS4, and CS5 is "Fully". In CS1 user requirements were taken into consideration via utilising a user pool and conducting a set of interviews with users to pinpoint their problems and goals. In addition, heuristic evaluation was performed on similar projects to gain an understanding of user needs.

Whereas, CS2, CS3, and CS4 gathered user requirements via conducting contextual inquiry and interviews. In CS5 high level requirements were prepared by the product manager, then user testing was conducted via low fidelity prototypes followed by interviews with users that resulted in pinpointing users problems and goals.

Participant PT1 described requirement gathering process and the utilisation of user pool by stating that

> We picked a few [users] based on product manager recommendations, people [users] that were kind of good at communicating their needs I think we picked also based on the size [of business] so people [users] who did a lot of business and people who did little business because the design was you know was dependent on the frequency of use of this particular product so we had a couple of that was really selling like millions on Amazon.com and the others were starting up or were not doing that much, our approach to be actually able to capture both ends of spectrum.

Participant PT2 discussed his need to conduct contextual inquiry by stating that

> What I told management was, [...] I want to do research upfront about how they [users] actually think about these processes so one thing I wanted to do was watch what they [users] do [...] do they [users] have a note book next to them and they [users] diagram it out or do they [users] use MS Visio to diagram it out [...] and what are all the tiny little details that they [users] need to figure out and how do they [users] need to view them in order to see them all in one place and in order to manage these

> processes so we [UX team) did a bunch of research upfront before we even started designing.

Participant PT3 described the user requirements gathering process by stating that

> I would do interviews with stakeholders and end users I also used contextual inquiry right so I would go and I would sit with people [users] occasionally and I would actually do the job myself just to get firsthand experience.

Participant PT5 described the user requirements gathering process and the role of the product manager in it by stating that

> The product manager came over with sort of high level requirements so based on that we actually went on and did some initial validation so like I said we initially tested old products and we kind of identified what some of the problem areas were and then we went through and did some initial early concept designs and then we actually tested those with users so from that we kind of got a sense of what they needed and we kind of validated a little bit the requirements that was given to us and we kind of exchanged backwards to product management and sort of worked through the process that way.

Practice A2.2- Performance of Relevant Practices

The rating for practice A2.2 for CS1 is "Fully" since a number of practices were conducted to include user requirements into the development process. These practices were: first, user requirements were used in task analysis. Second, a user experience vision was designed that includes high level user goals, description of the product's UX and the product's navigation model. Third, a number of Personas were designed. Fourth, the functional product owner and the usability product owner jointly specified the user stories and maintained the development backlog. Fifth, usability testing sessions were conducted with high fidelity prototypes. The rating for practice A2.2 for CS2 and CS3 is "Fully" since user requirements were included into the development process via including user requirements as features in the product backlog and then translating these features into a set of user stories.

The rating for practice A2.2 for CS4 is "Fully" since information regarding user requirements were written in a basic vision document, placed on a shared portal, that described the project and different types of users in-

volved and their characteristics. Extreme scenario based design was also used to include information regarding user requirements into the development life cycle.

The rating for practice A2.2 for CS5 is "Partially" since although use cases were prepared by the UX team, task analysis and user flows and some initial designs were made that reflect user needs as well as design specifications, yet the engineering team acquired this specifications and did not communicate or allow the UX team to iterate on that design.

Participant PT3 discussed the practices performed in order to include user requirements into the development process by stating that

> I would write user stories and hand them out at that time I was not a product owner but I would write user stories and I would have the product owner put them in a backlog.

Participant PT5 discussed problems with performance of activities related to the inclusion of user requirements in the software due to differences in the working methods between the engineering team and the user experience team by stating that

> We initially came up with a set of use cases and so we actually started designing but the thing is I think the way that the project went with engineering is that the Agile is new to them as well so they sort of fall back into their waterfall methods where they wanted everything upfront from us and so we told them it was not possible because we could not do that in the time they were giving us so we took some initial cases that I think we all agreed we are going to work on it and implement and then we provided some, we did the design work, we did some reviews and then they asked us for basically a spec and we provided the spec but once they got the spec they actually never came back to us and asked us questions about it or anything like that so they basically went on and implemented it.

Thus the collective rating of the Performed Process Attribute for CS1, CS2, CS3, and CS4 is "Fully" and "Unknown" for CS5 since the rating for practice A2.1 was "Fully" whereas that of practice A2.2 was "Partially". Thus it was not possible to decide on the combined rating with the assessment guidelines provided by UMM-HCS documentation.

B.1-Quality in Use Awareness Attribute

Quality in use awareness attribute is composed of three practices: practice B1.1- Quality in use training, B1.2- Human centred methods training and B1.3- Human system interaction training.

Practice B1.1- Quality in Use Training

The rating for practice B1.1 for CS1 and CS3 is "Fully" since formal usability training sessions is conducted to staff via the usability product owner in CS1 and lead UX designer in CS3. In CS3 staff were asked to read and discuss particular parts of usability books in order to raise usability awareness. The rating for practice B1.1 for CS2 and CS4 is "Partially". Practice B1.1 in CS2 was given a rating of "Partially" since training was conducted to staff but it was focused on raising awareness on the difference between usability and utility. Practice B1.1 in CS4 and CS5 was given a rating of "Partially" since only informal form of training is conducted that introduces usability and usability practitioners' tasks.

Participant PT1 mentioned that there was a lot of interest from staff to attend the training conducted on usability related aspects. Participant PT1 stated that training attendees involved

> [...] The product manager, the development team, the documentation team and sometimes even sort of senior management.

Participant PT2 discussed the details of the usability training by stating that

> We [UX team] did a large education effort around the difference between usability and utility.

Participant PT3 discussed the request he made to staff to educate themselves on usability via reading UX books and commenting on them. He stated that

> Like here is your home work [to read particular UX book] read this and they would read it and we would get comments and I would go like I want comments the next day.

Participant PT4 described informal usability training conducted by stating that

> I would say come bring your lunch here this time and I will give you an introduction to [...] what is usability and what are

some of the usability practices and how are they useful.

Practice B1.2- Human-Centred Methods Training

The rating for practice B1.2 for CS1 and CS3 is "Fully" since formal user centred methods training sessions were conducted to staff via the usability product owner in CS1 and lead UX designer in CS3. In CS1 the training involved conceptual modeling, user mental modeling, and tasks modeling. In CS3 it covered different UCD techniques and the design studio, the specific technique used by PT3 to integrate Agile development processes and UCD. The rating for practices B1.2 for CS2 is "Not achieved" since no training was conducted to staff to raise awareness on UCD methods or human system interaction. The reasons behind this are: first, the presence of a central UX team that is responsible for serving UX needs and that is composed of employees who all have a masters degree in HCI, i.e., conducting user research, usability tests, etc. This central team is composed of 50 UX practitioners and as a result the need for training staff is subsided by the presence of these available and highly qualified UX team who is trained via three months of mentoring, guidance from UX manager and training Internet page with UX training materials and process guidelines. Second, ad hoc training is conducted to product manager and development teams that is focused on how UX is integrated into the Agile development life cycle rather than discussing usability awareness, UCD methods or human system interaction. The rating for practice B1.2 for CS4 and CS5 is "Partially" since only informal form of training is conducted that discussed different UCD techniques.

Participant PT3 discussed monthly training on user centred methods that resulted in educating developers on how to perform some of these methods on their own, he stated that

> I gave presentations we do monthly we do an all hands on meeting [...] and explain why I was there [in the company] and here are some of the techniques that I use and why is it important and so forth [...] once we started the design studio after a while the team wanted to get involved they were like why can't we go down and talk to users [...] so we gave them [developers] a set of interview guidelines [...] that worked out well.

Participant PT4 described UCD activities covered in user centred methods training by stating that

> Just basic stuff like what is UCD things like user descriptions,

> personas, data gathering techniques like why would you want to go to the site why would you want to do a site visit, contextual inquiry type stuff, what are the different type of evaluations like using actual lab based user testing versus like expert.

Participant PT5 described topics covered in user centred methods training by stating that

> [...] what is UCD process and we go through the analysis, design, verification and all the different steps involved and [...] what we try to message is that we can work in Agile process like a lot of the steps in UCD kind of align with Agile and then the other part of it for them to understand what is it that we are doing like for a lot of the research that we are doing what is user testing [...] what we do we go through during the design process how do we verify that we are actually meeting the needs of the users.

Practice B1.3- Human-System Interaction Training

The rating for practice B1.3 for CS1 is "Fully" since formal human system interaction training sessions is conducted to staff via the usability product owner. The rating for practice B1.3 for CS2, CS3, CS4, and CS5 is "Not achieved" since no human system interaction training sessions is conducted to staff. The reasons behind lack of human-system interaction training for CS2 are the same as those discussed for practice B1.2.

Participant PT2 discussed how ad hoc training is conducted to product managers and Scrum teams on how UX is integrated into the Agile development life cycle by stating that

> And my team ultimately educates each Scrum team kind of as they are assigned to them, so new Scrum teams are created or new PMs come to our world and every time you are assigned to a new team or every time a new person come in the company there is education a sort of on ramp that you have to put them through to teach them how UX does it into Agile and how do they need to accommodate it, like I am going through that now with the new PM.

Participant PT2 expressed the lack of formal training to developers on usability, UCD methods, human system interaction, user consideration or context of use by stating that

> They [developers] do not [take training] and it probably causes a lot of problems [...] my manager years ago tried to create something [staff training] and then it was like half implemented and it did not get into new hire training, our new hire training.

Thus the collective rating for Quality in Use Awareness Attribute for CS1 and CS3 is "Fully" whereas it is "Not achieved" for CS2 and "Partially" for CS4 and CS5.

B.2- User Focus Attribute

User Focus Attribute is composed of two practices: Practice B2.1- User Consideration Training and Practice B2.2- Context of Use Training.

Practice B2.1- User Consideration Training

The rating for practice B2.1 for CS1 is "Fully" since formal user consideration training sessions is conducted to staff via the usability product owner. Whereas it is "Not achieved" for CS2 for the same reasons indicated earlier for practices B1.1, B1.2, and B1.3. Practice B2.1 rating is "Largely" for CS3 since user consideration is touched upon in the monthly training sessions conducted by lead UX designer to staff and personas are used to raise staff awareness to the importance of considering the needs of end users when developing the system, however, users are collocated so they are very accessible and thus other alternatives are available for staff to consider user needs while developing the system. Practice B2.1 rating is "Partially" for CS4 and CS5 since only informal form of training is conducted that discussed user consideration. Participant PT3 discussed the utilisation of personas by stating that

> I will put them [personas] on the wall or something so that the team can read them and you know get the information.

Practice B2.2- Context of Use Training

The rating for practice B2.2 for CS1 is "Fully" since formal context of use training sessions is conducted to staff via the usability product owner that discusses user mental modeling. Whereas it is "Not achieved" for CS2 for the same reasons indicated earlier for practices B1.1, B1.2, and B1.3. The rating is "Largely" for CS3 since monthly training sessions are conducted by the lead UX designer to staff that discuss a number of issues including context of use. Practice B2.2 rating is "Partially" for CS4 and CS5 since only informal form of training is conducted that discussed context of use.

Participant PT3 discussed details on context of use training conducted to staff by stating that

> I did present and show kind of the triangle if you will of business needs, user needs and technical needs which are the developers, I mean developers have a tendency to drift to technical.

The collective rating for User Focus Attribute for CS1 is "Fully", "Not achieved" for CS2, "Largely" for CS3 and "Partially" for CS4 and CS5.

C.1-User Involvement Attribute

User Involvement attribute is composed of four practices. Practice C1.1- Active involvement of users, C1.2- Elicitation of user experience, C1.3- End users define quality in use, and C1.4- Continuous evaluation.

Practice C1.1- Active Involvement of Users

The rating for practice C1.1 for CS1, CS2, and CS3 is "Fully" since there was early and continual user involvement throughout the development life cycle. CS1 achieved early user involvement via utilising a user pool to conduct interviews to gather user requirements while CS2 achieved it via involving users in ideation phase via brainstorming sessions, conducting contextual inquiry and interviews to identify user needs, and conducting low fidelity prototyping sessions. CS3 achieved early user involvement via conducting contextual inquiries and interviews. Moreover, users were involved in the design phase via the design studio.

Continual user involvement is embodied in CS1 and CS2 via continual usability testing sessions that were conducted with high fidelity prototypes. In CS3 continual user involvement was facilitated due to the colocation of users and the type of software developed by the company since the company develop software that facilitate users' work, so users were very cooperative in usability testing sessions. The rating for practice C1.1 for CS4 is "Largely" since although early user involvement was maintained via interviews and contextual inquiry yet continual user involvement depended on an on site customer who acted as a user proxy, hall way testing and occasional usability testing sessions that occurred with users on site.

As for CS5 the rating was "Partially" since although early user involvement was maintained via low fidelity prototype testing and interviews yet there was an absence of continual user involvement.

Participant PT2 expressed the early and continual involvement of users in

brainstorming sessions, early design phases and throughout the development life cycle by stating that

> I think there are three places I would say that I try to get users involved, one is kind of the phase of ideation. So what are the ideas what do we want to build, do those ideas resonate with users, what are the needs that they have right that we are not filling and what might we do to serve their needs [...] we will usually do some concepts so might be in the form of a story board something very low fidelity [...] and once we get validation [...] then we build a very high fidelity prototype [...] and we do RITE testing so like I typically run two to three users and then through usability testing with high fidelity prototype and then we will take a break for may be half a day or a day and we will regroup as a team and talk about the problems that we [UX team] saw and what we want to try for solutions and the designers will go and iterate on the design and then the next day or the next two days will do another set of users and we just keep basically doing that until we get to the point in which we have resolved all of major problems.

Participant PT3 discussed users' cooperation and ease of accessibility by stating that

> With our users collocated it is fairly easy to go down and see what they do [...] it is a huge advantage [...] another kind of twist to what I do is that I design enterprise business applications so I do not design for the public I design for people that have to use this software for work [...] any usability changes that I make makes their lives a lot easier.

Participant PT4 discussed the continual involvement of on site customer who acted as a user proxy in the development process by stating that

> So basically it was throughout but not with all the people that I would like to have been there, so with this project it was OK because our client / or customer contact was basically had experience working in the shop so he knew everything that was going [...] and he had been promoted to the manager position so he knew all the contextual issues.

Participant PT5 discussed the lack of continual user involvement and its reasons by stating that

> We did some initial testing upfront and then after that it basically we were not able to really do much testing after that because of the back and forth with the engineering team and the fact that we were struggling to do, to get all the design done and sort of give them [engineering team] what they wanted and the format they [engineering team] wanted [...] we did not have the opportunity to evaluate with users.

Practice C1.2- Elicitation of User Experience

The rating for practice C1.2 for CS1, CS2, CS3, and CS4 is "Fully" since the design was shown to stakeholders and they were allowed to perform simulated tasks in case of low fidelity prototypes or tasks in case of high fidelity prototypes. This occurred via early and continual usability sessions. For CS5 the rating was "Partially" since although user experience was elicited upfront via low fidelity prototype testing yet there was an absence of any form of user experience elicitation throughout the development process.

Participant PT1 discussed the method used for eliciting user experience by stating that

> They [users] gave their feedback usually we would have them go through the steps and do the tasks [...] and the feedback would pretty much be verbal [...] and we would take that [feedback] back and roll it into we would make changes based on that essentially so we would roll it as fast as possible and go back and modify our wire frames go back and change what we are planning to do to make sure we incorporated that input.

Participant PT2 quote on practice C1.1 signifies the early and continual efforts for eliciting user requirements.

Participant PT3 discussed the method used for remote usability testing by stating that

> What we do is we give [...] a sampling of users access to our QA environment [...] and we start collecting feedback.

Participant PT4 discussed how he was keen, as the usability engineer, for users to perform simulated tasks on the software, by stating that

> Basically we wanted them to perform tasks using the software, we did not want them to just look at it.

Participant PT5 discussed the implications of lack of elicitation of user experience on the end product by stating that

> There was like a major problem with the system [...] basically users were unable to purchase completely successfully so they [engineering team] had to pull back the product and spend a couple of months trying to fix it.

Practice C1.3- End Users Define Quality in Use

The rating for practice C1.3 for CS1 and CS3 is "Largely" since measures of usability were derived indirectly from users via the results of user requirements gathering and results of usability evaluations. CS1 and CS3 derived usability measures from the results of heuristic evaluations. The rating for practice C1.3 for CS2 and CS4 is "Fully" since measures of quality in use were mainly derived indirectly from users via the results of user requirements gathering and results of usability evaluations. In CS5 the rating was "Not achieved" since measures of usability were driven from general usability guidelines rather than from users.

Participant PT3 partially defined acceptable usability measures via utilising the ergonomics bench marks for human perception. He stated that

> The ergonomics bench marks for like human perception [...] so I look at things loading too fast did the user even notice that anything happened let us slow it down let us put a pause in there so I look and I analyse the UI at that kind of micro level from my background.

Participant PT4 described the method used for deriving acceptable usability measures from end users by stating that

> [...] Interviews with the clients when we were visioning or creating the vision for the system I mean he [user] would tell us [...] what he wanted, what were the problems with the activity and what they want to accomplish and then based on that we would determine the efficiency matters.

Participant PT5 described how he derived acceptable usability measures via general usability guidelines by stating that

> I think they are more general guidelines than anything else based on this general usability practices so you know I do not think we had I would not say we had a clear sense matrix I would not say, there was nothing defined by management

> either so basically going with what we thought good usability practices.

Practice C1.4- Continuous Evaluation

The rating for practice C1.4 for CS1, CS2, CS3, and CS4 is "Fully" since continual testing occurred via high fidelity prototypes in order to gather feedback. The rating was "Partially" for CS5 since although early user involvement was maintained yet there was an absence of continual user involvement as it is clear from participant PT5 quotes on practice .

Participant PT1 described the continuous user evaluation by stating that

> We continued to get feedback from our users so we kept doing you know [...] at every sort of stage when we had something working we would take it back to one or two of those 6 people [user pool], so it was quite consistent, continuous the feedback.

Participant PT2 quote on practice C1.1 signifies the early and continual efforts for eliciting user requirements. Participant PT2 described the continuous evaluation and involvement of developers in usability testing sessions by stating that

> It is very iterative, we [developers] are getting their [users] feedback and we are adapting in real time.

Participant PT2 also expressed the importance of frequent usability testing that is attended by developers, he stated that

> Every time we [UX team] run a test the team participates, watches, observes and talks about the results and about the changes and what is possible I mean it is part of the iteration process too [...] getting information from me [participant PT2] is second hand, hearing the users express this frustration and needs is very powerful.

Participant PT3 quotes on practice C1.1 and C1.2 signifies the efforts towards continual evaluation. Participant PT4 discussed the timing and methods used for usability evaluations by stating that

> In each iteration the usability person is doing the design for the next iteration, they are helping the developers implement their design from the last iteration and they are also doing the usability evaluation for anything that was implemented and delivered in the last iteration, so basically at each iteration you are doing some sort of usability testing.

Participant PT5 expressed the lack of continuous evaluation on users by stating that

> We did internal iterations but nothing validated with users [...] we did not have the opportunity to evaluate with users.

The collective rating for User Involvement Attribute for CS1, CS2, CS3, and CS4 is "Fully" and "Partially" for CS5.

C.2-HF Technology Attribute

HF Technology Attribute is composed of three practices. Practice C2.1- Provide appropriate human centred methods, C2.2- Provide suitable facilities and tools, and C2.3- Maintain quality in use techniques.

Practice C2.1- Provide Appropriate Human Centred Methods

The rating for practice C2.1 for CS1, CS2, and CS3 is "Fully" and "Largely" for CS4 since various methods were used for elicitation of user input at all stages in the development life cycle. For CS5 it was "Partially" due to the limited activity in regards to elicitation of user input in all stages in the life cycle as stated in practices C1.1, C1.2, and C1.4. The methods were interviews (CS1, CS2, CS3, CS4, CS5), contextual inquiry (CS2, CS3, CS4), usability testing (CS1, CS2, CS4, CS5), and remote usability testing (CS1, CS2, CS3). Although, CS4 conducted some testing on users, yet most of the testing were conducted on an on site customer who acted as a user proxy as well as conducting hallway testing and heuristic evaluation as discussed in quotes and description of practice C1.4.

Participant PT2 stated that the UX team established a systematic approach for choosing appropriate UCD methods for the different projects by stating that

> My [participant PT2] research team we made [...] our internet page made [...] a one page document that describes a method: when we use it, who the go to person on the research team who knows the most about that method, and then an example project that use this method [...] we are trying to educate our team that usability testing is not the only method and that you need to [...] think what is the research question that I am trying to solve, and what is the right method to answer this question.

Participant PT2 described how contextual inquiry was used for early elicitation of user input for the design of an approval process editor by stating that

> We need to know what their [users] mental model is and since every organisation has some kind of approval process going on we can actually have them [users] work through that process and how they [users] develop an approval process in front of us and in that way we can say what tools do they use like I said do they report it do they pick up a note book do they open up a different program and that is something that I have to be there in real time to see so in that case like there is contextual inquiry is just the right method for the problem that I had in hand.

Moreover, participant PT2 considered using several human centred methods and decided to resort to remote usability testing. He discussed his reasons for preferring remote usability testing by stating that

> But almost always we do remote usability testing so the participant does not come to the lab, that enables first of all, you do not have people wasting a bunch of time trying to find your building, etc. or being on a train that just broke down or something like that. Also by doing remote usability I can conduct sessions with anyone in the world right I am not limited to people in my local area.

Practice C2.2- Provide Suitable Facilities and Tools

The rating for practice C2.2 for CS1, CS2, and CS3 is "Fully" and "Largely" for CS4 and CS5 since suitable facilities and tools were provided for quality in use techniques, for example, prototyping tools like Visio (CS1), Power Point (CS4) or Dreamweaver (CS3), video recording tools (CS2, CS4, CS5), laptops (CS4) and facilities, for example, dedicated usability labs (CS2, CS5). In CS4 these tools were provided if they are reasonably priced. Participant PT1 declared that management was fully cooperative in regards to any needed tools or facilities by stating that

> When I joined they [management] asked me if I needed any of that [facilities] and my response was no because really I think that the best method is for [...] so most of the time it is better to be where the user is and get input from them and their environment.

Participant PT1 declared also that management were cooperative in case the UCD team required any tools

> There was no problem in getting anything [tools].

Participant PT2 discussed the presence of a number of usability labs by

stating that

> We have five dedicated usability labs [...] we have two labs which were set in a very traditional format where we have the one way mirror where the user would sit on the other side of the mirror from you and it was like the facilitation room or the observation room.

Participant PT3 discussed his reasons for not needing a dedicated usability lab by stating that

> In my professional opinion that you can get 80% of what you need with guerrilla tactics, observing users and talking to them and listening to what they are saying or what they are not saying [...] a lot of that stuff [dedicated usability labs] is to me are expensive refinements.

Participant PT4 discussed management support to UCD practitioners in regards to needed tools by stating that

> They [management] provide me with tools as long as it is reasonably priced [...] they provide me with laptop so as I can do user testing and take it to user sites and do testing you know and do data collection and that sort of thing you know if I need to do I mean everything is kind of I ask for it if I need it.

Participant PT5 discussed management support in providing facilities and tools by stating that

> We actually have our own usability lab so that was one thing that was nice so we had an observation room and a testing room back then [...] I think tools were not really an issue I mean we had all the recording tools and equipment that we needed in order to record the sessions.

Practice C2.3- Maintain Quality in Use Techniques

The rating for practice C2.3 for CS1, CS3, and CS4 is "Largely" since UCD methods and techniques were reviewed for suitability and results of review were shared with the development team. CS3 improves UCD methods and techniques via staff attendance to different conferences and bringing new techniques and modifications to techniques. The rating for practice C2.3 for CS2 is "Fully" since UCD methods and techniques are reviewed for suitability and for ensuring that state of the art UI technologies are appropriately used in new systems development. This occurs during

mentoring of new UX practitioners or during the creation of new training material. As for CS5 the rating was "Partially" since some review occurred during retrospectives.

Participant PT2 discussed methods and timing used for reviewing UCD techniques by stating that

> Well I mean I think that when we end up reviewing it [UCD techniques] is when you are like mentoring somebody or creating a new training material or something like that but as a whole you know it comes out of discussion. Certainly like I just told you about that lab change right that became when I started I had a microphone I had people coming in to the lab you know and it evolved into the lab where we reconstructed our labs to have more of them in order to conduct it [testing] remotely so yes they are getting reviewed and altered and evolving over time.

Participant PT3 discussed the utilisation of state of art UCD techniques via UX team staff and management attendance to different conferences, he stated that

> The design studio we picked up at a conference at San Francisco, on seeing a presentation of design studio process we immediately realised that that was worth trying out with an Agile team [...] We had a couple of our senior management too are really interested in what I do and understand it and are on my side so I mean they would go to conferences and come back with ideas too.

Participant PT4 discussed continual review of UCD techniques by stating that

> I was doing kind of continual data collection and reflection on what was working and what was not working and kind of tweaking what we were doing as we went along.

The collective rating for HF Technology Attribute for CS1, CS2, and CS3 is "Fully", "Largely" for CS4 and "Partially" for CS5.

C.3-HF Skills Attribute

HF Skills Attribute is composed of three practices. Practice C3.1- Decide on required skills, C3.2- Develop appropriate skills, and C3.3- Deploy appropriate staff.

Practice C3.1- Decide on Required Skills

The rating for practice C3.1 for CS1 and CS5 is "Not achieved" since there is an absence of a systematic procedure in place to identify essential UCD competencies and plan their availability so as to ease multidisciplinary design solutions.

The rating of practice C3.1 for CS2 is "Fully" since there is a clear procedure in place for pinpointing the essential UCD competencies and planning their availability so as to facilitate multidisciplinary design solutions. CS2 had a clear method by which UX team members are hired to ensure competencies. This method involved preparing a long list of competencies, conducting 10 interviews per UX practitioner, asking UX practitioner to conduct a presentation of their historical work, perform a home work assignment to assess their ability to deal with usability problem, and conduct a mock-up usability session. The rating of practice C3.1 for CS3 and CS4 is "Largely" since PT3 and PT4 are responsible for identifying the required competencies and planning for their availability in order to facilitate multidisciplinary design solutions.

Participant PT2 described the hiring process by stating that

> We have a long list of competencies and then we have a very long hiring process which involve sun screens and a portfolio presentation of their historical work they do a home work assignment where we assess their ability of thinking through a usability problem they have to do a mock-up usability session where they conduct a usability session with somebody from our team and then they go through about 10 interviews

In addition, participant PT2 described the hiring process as very formal due to the following reasons

> It [hiring process] is extremely formal and everyone like each interview has defined interview topics and they have a list of questions to go from and you know we have tracking tools for their [interviewers] feedback on the candidates.

CS2 only hired staff with a masters degree in HCI as started by participant PT2

> We also hired only people who primarily that come from masters degree in HCI program [...]they know in theory how it is supposed to work.

Moreover, CS2 clearly distinguished between designers and researchers in order to maintain design quality as stated by participant PT2

> At Salesforce they [management] do not combine roles of design and researcher [...] to keep things honest, a designer will never be the one assessing whether users can use their designs.

CS2 also had a clear process for allocating UX team members to the different Scrum teams. This method involved allocating one UX team member to a maximum of two projects and using the Office Hours (OH) for projects that do not have an assigned UX team member where UX team members allocate two hours weekly for helping Scrum teams with no assigned UX resources.

Participant PT3 described his responsibility and method for identification of required competencies in UX team by stating that

> I write all the job descriptions and I interview the candidates [...] when I built the team what I did is I know that my skills are leading and organising, I have really good interaction design skills, information architecture skills, what I lack is graphic art skills so I am really really strong early at the design process,[...] so I wanted somebody who was detail oriented who had good visual design skills set, [...] what I looked for was complementary strength rather than redundancy and that would be that approach work and I would certainly use it again.

Practice C3.2- Develop Appropriate Skills

The rating for practice C3.2 for CS1 is "Largely" since a number of ways are utilised to develop appropriate skills in UCD staff. Those ways were training, attending conferences and job experience. However, attending conferences is only a privilege offered to usability product manager. The rating of practice C3.2 for CS2 is "Fully" since a clear and extensive procedure is followed to develop appropriate skills in UX team via training, attending conferences, and job experience. CS2 develops UX team skills via various methods: first, mentoring program where every new UX researcher is assigned a mentor for 90 days who supports them in planning and conducting UX research sessions and educates them in communicating with Scrum teams. Second, training web site with UCD materials and process guidelines. Third, feedback provided by UX manager. Fourth, attending conferences. Fifth, attending informal training sessions. Sixth, learning via job experience. The rating of practice C3.2 for CS3, CS4, and CS5 is "Partially" since skills for UCD staff are developed via job experi-

ence (CS3, CS4, CS5), attending conferences (CS3, CS4, CS5), attending off site or ad-hoc training(CS5), and self development (CS3).

Participant PT2 discussed the mentoring program duration and activities involved by stating that

> Training most of it happen through mentoring, so every researcher [UX researcher] if they are new [...] is assigned a mentor [...], buddy for 90 days, [...] the mentor sits in on any research session to help them [new UX researcher] plan the research, goes to some of their meetings with them, teaches them how to handle teams [...] they [new UX researcher] will get trained on for sure how to use the lab, how to conduct a session with the lab. All the details around running a lab session [...] legal disclaimers that you need to give [...] and then there is the way that we facilitate and take notes and manage our teams and our setting and the expectations around you [new UX researcher] [...] and then the expectations on exactly how do you do your reporting on your results because that is extremely standardised [...] then they have a peer reviewer for any material that they are sending out, any written material.

Participant PT2 discussed the training website by declaring that

> We also like have a training Internet page where we have a lot of materials and process guidelines.

Participant PT2 elaborated on learning through job experience by stating that

> They [new UX employees] do learn Agile, like I said, they do not know the intricacies of UX in Agile other than through mentoring and through I guess just trial and error and learning as they go.

Participant PT3 discussed sources for developing skills of UCD staff by stating that

> Generally conferences and just knowledge sharing between us [UX staff] [...] I think every UX person does this when I am home I read a book and I talk to people and I network.

Participant PT5 described the development of skills via training as an ad hoc activity by stating that

> It was basically sort of ad hoc so somebody were interested in getting training for specific things they would just raise it to management you know and they would make a decision whether to send a person or not, it worked that way with conferences and it worked that way with training classes as well.

Practice C3.3- Deploy Appropriate Staff

The rating for practice C3.3 for CS1, CS2, CS3, and CS4 is "Largely". For CS1, CS2, and CS4 this was attributed to involvement of skilled UCD staff in all stages of development as and when required. In CS1 and CS2 due to the amount of projects assigned to UCD practitioners, they are more involved and effective in all stages of development of projects where UCD has a high priority. In CS2, one UX team member is allocated to a maximum of two projects where UCD has a high priority, this leaves some Scrum teams with no UX practitioner, however, Office Hours (OH) are used for those projects where UX team members provide two hours weekly to help those Scrum teams. For CS3 this is attributed to business needs that sometimes lead to prioritising functional features over UX features. For CS5 it is "Partially" since UX team was only involved in the early phases of development as discussed in participant PT5 quotes on practice A2.2, C1.1, C1.2, and C1.4.

Participant PT2 expressed the effect of workload on UX researchers by stating that

> Unfortunately because of resourcing it is not quite we are not quite at a level where we can may be do the depth of involvement that would be ideal.

Participant PT3 described his efforts to include and prioritise UX features into the backlog by stating that

> My job included wining over the product owner [...] I spent more time convincing the product owner [...] that this stuff [UX] is important right [...] the tendency for product owners that have come up through that [business] path is business requirements [...] they choose the shortest path because they are very time and budget conscious [...] I was always kind of fighting. It was me and the product owner, me trying to get myself prioritised and him going no, business needs, drop that, and going back and forth.

Participant PT3 described the role of the usability engineer in the develop-

ment life cycle by stating that

> As I mentioned before at any given iteration the usability engineer is basically doing three things: is working on the design for the next iteration or iterations, doing an iteration of what was delivered previously, and working with the developer to make sure that designs are being translated to implementation correctly.

The collective rating for HF Skills Attribute for CS1, CS3, and CS4 is "Largely", CS2 is "Fully", and CS5 is "Partially".

D.1- Integration Attribute

Integration Attribute is composed of three practices. Practice D1.1-Integrate HF processes, D1.2-Facilitate interface between HF and the organisation, and D1.3-Use appropriate representations.

Practice D1.1- Integrate HF Processes

The rating for practice D1.1 for CS1 and CS3 is "Partially" since although quality assurance team is part of the development team and they are very cooperative and responsible for reviewing both functionality and usability aspects of the software yet there is no integrated process for UCD practitioners and quality assurance practitioners. Practice D1.1 rating for CS4 and CS5 is "Not achieved" since there is an absence of an integration process between usability engineers and quality assurance team.

Participant PT1 described the integration with the quality assurance team by stating that

> They [Quality Assurance] were actually really good I always find that QA, the quality assurance guys, they were actually, they really help the product manager especially the usability aspects of it because they constantly kind of testing and they you know because they spend so much time with the testing they have a very good sense of things that are sort of inconsistent [...] and they will question why they will bring those to your attention quite frequently so I find that very valuable.

The rating for practice D1.1 for CS2 is "Largely" since although there is a clear process in place to ensure quality, that classifies bugs according to their severity into a number of categories, however, some QAs try to release the product by changing the severity of usability bugs rather than reporting them. However, some efforts are exerted by UX researchers

to prevent such incidents from occurring. Participant PT2 expressed this situation by stating that

> The mandate is that we will never release anything with a P1 or P2 [priority 1 or priority 2] bug [...] then the reality is when we get close to release a lot of the bugs get reassigned to a new priority like a P2 suddenly becomes P3 very conveniently! So the product is then releasable. So we have done a lot of work recently to try to enhance like if one of my bugs gets de-prioritised the person [quality assurance] has to explain why and then I am notified through an automated system so I could be like hey actually that is a P2, change that back. So actually so yes we do have a process but it is not perfect.

Participant PT3 discussed the close collaboration with the QA team by stating that

> The QA team that has worked with me is very well versed in looking at this thing [usability issues] as well [...] these guys' attention to detail is just phenomenal so I have kind of allies that help me with that along the way.

Participant PT4 expressed problems in collaboration with the QA team by stating that

> We also did have problems because what would happen is that the QA department would be doing their testing at the same time that the usability engineer might be doing user testing and if the QA department verifies some functionality and later it turns out that we need to do some additional changes because of the user testing results then the QA department would then have to go back and do some testing again [...] we did not have anything formal in place.

Participant PT5 expressed problems in collaboration with the QA team by stating that

> The QA was with the engineering team [non collocated with UX team] and again it [process] was very isolated so even when we were trying to explain what the designs were so that the QA can test against it I do not think that happened either so you know we kept the engineering team sort of kept the QA team sort out of the cycle [...] we did not do any reviews with them so I am pretty sure they [QA] did not fully understand

the designs either.

Practice D1.2- Facilitate Interface between HF and the Organisation

The rating for practice D1.2 for CS1 and CS3 is "Largely" since efforts were exerted by the UCD team to facilitate the interface between UCD practitioners and the organisation via using a language and working methods that are appropriate to successful interaction with other departments. CS3 benefited from development team's involvement in the design studio and also conducted monthly training sessions as illustrated from practices B1.1, B1.2, B2.1, and B2.2 and their accompanying quotes and description. The rating for practice D1.2 for CS2 is "Fully" since UX team conducts usability awareness training and uses a language that is simple enough to communicate successfully with other departments. Moreover, a UX team member is assigned to a maximum of two projects with high UCD priority and the office hours schema is used for projects that does not have an assigned UX team member where UX team members provide two hours of their time per week to assist Scrum teams that did not have assigned UX resources and since the UX team is composed of 50 members then this provides suitable support to different Scrum teams. This allows the UX team member to be collocated with the Scrum team to instantly answer any questions or issues raised by the development team. As for CS4 the rating of practice D1.2 is "Partially" since efforts were exerted by the usability engineer to facilitate the interface between UCD practitioners and the organisation via conducting informal training sessions, yet some problems existed with QA team as discussed in practice D1.2. CS5 is "Not achieved" since there is an absence of any formal process in place to facilitate the interface between UX team and the organisation.

An example that illustrates problems in CS1 with communicating with developers and managers and efforts exerted to overcome these problems via training sessions is expressed by participant PT1

> The terminology often I have had people [developers and managers] sort of say you know I really do not know what you are talking about so the word that you use means nothing to me so often I had that sometimes, you tend just because you know the term you tend to use them [...] I think mostly I try to role that into the training [...] this is what we did you know here is the conceptual model, what is the conceptual model, what is the domain model, why are they different? What is persona? Why do we have a persona? You know, so basically trying to educate them and share kind of the technique and all of that

stuff with them.

These efforts resulted in resolving initial friction with functional product manager as participant PT1 declared

> I think there was friction definitely in the beginning [...] because it was very hard for the product owner [...] to say OK we will do this the UX way [...] but I think that with the passage of time it only got better [...] and [...] the other product owner because he kind of became the strongest proponent of UCD by the end of the project definitely.

Participant PT2 discussed the efforts exerted by the UX team to facilitate the interface between UX team and the organisation via conducting training by stating that

> We [UX team] did a large education effort around the difference between usability and utility.

Participant PT2 discussed the continuous communication between UX team members and developers as a result of colocation by stating that

> The designers are there to answer questions right [...] but with Agile the team is so small there is a lot of discussion back and forth so you know if there is questions so that is when the designer is sitting at the developer desk working on it with them.

Participant PT3 discussed the efforts exerted by the UX team to facilitate the interface between UX team and the developers via sharing results of users interviews by stating that

> One of the things we did though is share results with the team [development team] as soon as we got them so If I went out and did even one user interview I would come back to the team immediately after the interview and assemble them and say hey this is what I found or the very least the next Scrum at the next stand up I would share the results with the team right so they learned as I did which I thought were extremely valuable so there was no long reports or stuff like that for them to have to find and read and digest.

Moreover, participant PT3 discussed the role of the design studio in creating a common team vision by stating that

> The biggest side of the design studio is that everybody walks out of there you know everybody comes in a designer but when

> they walk out they walkout with a unified vision of what the product is, [...] everybody walks out of there with their own perspective of what needs to be done.

Practice D1.3- Use Appropriate Representations

The rating for practice D1.3 for CS1 and CS3 is "Largely" while the rating is "Fully" for CS2 and "Partially" for CS4 and CS5 since efforts were exerted to represent user requirements and system changes originating from user involvement in a way that is understandable by developers via including user requirements and system changes as stories in the backlog (CS1, CS2, CS3, CS5), using Visio to represent user requirements and system changes (CS1), developing a number of personas (CS1, CS3), including user requirements in fully interactive prototypes that clearly specifies usability measures (CS2), power point (CS4), scenarios and claims (CS4), use cases (CS5), task analysis (CS5), user flows (CS5), and in case of need of clarification, face to face communication occurred between designers and developers to represent user requirements and system changes (CS2, CS3). CS4 was rated as "Partially" since their representation method was considered to be heavy on documentation.

Participant PT1 described the presentation of requirements by stating that

> Well, the requirements I kind of present in lots of different ways so there is stories in the backlog and the task flows. What is the user trying to accomplish? Right, and what are the facts sort of and then comes the design so in the Visio diagram also personas, we did develop personas, for the user so that was kind of representation somewhat of the requirements so the requirements were kind of multifaceted representation of that and the manifestation into the UI kind of came later.

Participant PT2 discussed prototypes and face to face communication between designers and developers by stating that

> So the user stories [...] do not mandate how [...] and the user experience team visualizes the how through a prototype and then [...] the prototype is handed to the development team [...] it [prototype] is fully interactive, [...] there is no document that is handed over, it is just this prototype, so the development team eventually uses that, and the prototype is basically the specification [...], it is hard to misinterpret something when you are looking at like in front of your face [...], when you see it and feel it and you can interact with it it is a lot clearer [...]

> sometimes there are some cases like corner case or something that could possibly happen in the interface then the developer would go talk to the designer to talk about how to deal with that edge case that the designer may not have thought about.

Participant PT3 quote on practice D1.2 also provides insight on presenting user requirements and system changes originating from user involvement via face to face communication.

Participant PT4 discussed problems with the method used for representing user requirements and system changes by stating that

> One of the problems I had with my initial conception, my approach of excessive documentation is that it was far too much documentation heavy. The requirements, you have to write scenarios for everything and you have to write claims for all your features and you know put them together and organise them into an activity work flow and that sort of thing.

Participant PT4 was satisfied by the choice of power point for representing user requirements and system changes, he stated that

> We relied on power point because power point was kind of easy to develop and easy to show and communicate to other folks.

The collective rating for Integration Attribute for CS1 and CS3 is "Largely", "Fully" for CS2, "Partially" for CS4, and "Not achieved" for CS5.

D.2- Improvement Attribute

Improvement Attribute is composed of three practices. Practice D2.1- Ensure design feedback, D2.2- Change based on feedback, and D2.3- Timing of feedback.

Practice D2.1- Ensure Design Feedback

The rating for practice D2.1 for CS1, CS2, CS3, and CS4 is "Fully" since efforts were exerted to acquire feedback on design via conducting evaluations at all stages of the development life cycle as declared in analysis of practice C1.1, C1.2, C1.4, and C3.3 and participants' PT1, PT2, PT3, and PT4 quotes on them. The rating is "Partially" for CS5 as declared in analysis of practice C1.1, C1.2, C1.4, and C3.3 and participant PT5 quotes on them.

Participant PT5 discussed the lack of design feedback and its reasons by

stating that

> There was no time to go back in and re-verify, there was no time to test what was actually implemented, you know with the users, and there was no opportunity to do that.

Practice D2.2- Change based on Feedback

The rating for practice D2.2 for CS1 and CS3 is "Largely" since the development process encourages design changes based on the results of user evaluations. The rating for practice D2.2 for CS2 and CS4 is "Fully". In CS2 design changes are iteratively implemented throughout the development processes starting from the ideation phase until all design problems are resolved and is facilitated by using the RITE method. In CS4 the usability engineer declared that they overdone change based on feedback as it will be expressed in quotes. As for CS5 the rating is "Not achieved" since the development process discarded mostly design changes based on actual user experience.

Participant PT1 discussed the changes that can occur as a result of user evaluations by stating that

> Well, often I would go back and modify the wire frames and the user stories just sort of accommodate those results if they were sort of substantive on some of these actually user sessions would be included, we would have developers sit in and in that case what would happen is that before anything would happen the developers would go and make the changes so you know so that did happen but so it is mixed so if I think there is a little bit of complexity such like just changing a word here or there but if it is a little bit more complex like a new button or a new flow or a new set of fields or something like that then I would go back and change the story and the wire frame.

At some instances changes did not occur due to reasons stated by participant PT1

> Every now and then there would be like a technical constraint or you know some strong feeling by the product owner may be that it was not that high a priority [...] because it was time constraints or technical constraints that would kind of keep us from doing that.

Participant PT2 discussed changes that occurred as a result of users' feedback throughout the development life cycle by stating that

> Once we have some requirements we will usually do some concepts so might be in the form of a story board something very low fidelity that really is just enough detail to get across a concept or a scope and then what I like to do I call a concept testing [...] does this concept as we present it reasonate with you [user] does it provide value to you [...] really just trying to bring them [users] into that brain storming phase and making sure that the way we [UX team] are thinking about it [design] is the way that users are thinking about it [design] and once we get validation that we have a product worth building then we build a very high fidelity prototype [...] and in this test we do you know very specific task to gage whether the way we have designed it [software] is usable for them [...] and we do RITE testing so like I typically run two to three users and then through usability testing with high fidelity prototype and then we will take a break for may be half a day or a day and we will regroup as a team and talk about the problems that we [UX team] saw and what we want to try for solutions and the designers will go and iterate on the design and then the next day or the next two days will do another set of users and we just keep basically doing that until we get to the point in which we have resolved all of major problems.

Practice D2.2 was rated as "Largely" for CS3 since Participant PT3 sometimes struggled with the product manager in order to include and prioritise UX features into backlog as stated by participant PT3

> My job included wining over the product owner [...] I spent more time convincing the product owner [...] that this stuff [UX] is important right [...] the tendency for product owners that have come up through that [business] path is business requirements [...] they choose the shortest path because they are very time and budget conscious [...] I was always kind of fighting it was me and the product owner, me trying to get myself prioritised and him going no, business needs, drop that, and going back and forth.

Participant PT4 discussed how the usability engineers had overdone change based on feedback by stating that

> We were close to the user I would say but one of the problems we had I think in this project is that we did not do a good a job as we should have in terms of determining what prioritising

> changes. I mean the user would want something and we would say OK we will do it, like that without considering the impact, without analysing the request and communicating to users the impact of that request, but OK that is something that we got better at towards the end of the project.

Participant PT5 discussed lack of change based on actual user experience by stating that

> We did internal iterations but nothing validated with users [...] we did not have the opportunity to evaluate with users.

Practice D2.3- Timing of Feedback

The rating for practice D2.3 for CS1, CS2, and CS4 is "Fully" since the results of user evaluations in regards to user needs and software defects are fed into the design process as declared by participant PT1, PT2, and PT4 quotes on practice D2.2 while it was "Largely" for CS3 as discussed by PT3 quote on practice D2.3 and "Not achieved" for CS5 as discussed by participant PT5 quote on D2.2.

The collective rating for Improvement Attribute for CS1, CS2, CS3, and CS4 is "Fully" and "Not achieved" for CS5.

D.3- Iteration Attribute

Iteration Attribute is composed of three practices. Practice D3.1- Minimize risks by iteration of design, D3.2- Manage iteration of design solutions, and D3.3- Use design objectives to control iteration.

Practice D3.1- Minimize Risks by Iteration of Design

The rating for practice D3.1 for CS1 and CS3 is "Largely" and "Fully" for CS2 and CS4 since design is iterated using prototypes to increase the match between the final software produced and user expectations as illustrated earlier in quotes and description of practices C1.1, C1.2, C1.4, D2.1, and D2.2. Practices C1.2, C1.4, D2.1, and D2.2 for CS1 and CS3 revealed that early and continual iteration of design was conducted and results of usability testing sessions were fed into the development process. However, this practice was evaluated as "Largely" rather than "Fully" since there occurred some instances where design changes were discarded due to time or technical constraints. As for CS5 it is evaluated as "Partially" since iteration of design only occurred at early stages of the development life cycle as illustrated earlier in quotes and descriptions of practices C1.1, C1.2, C1.4, D2.1, and D2.2.

Practice D3.2- Manage Iteration of Design Solutions

The rating for practice D3.2 for CS1 and CS3 is "Partially" since Wiki is used to keep the latest designs (PT3) and notes resulting from usability sessions are recorded via Wiki (CS1). Participant PT1 stated that

> I keep like notes like on the Wiki sort of raw user input [...] I always keep every time that we have any session [...] so there would be sort of some record for that [...] sometimes you want to go back because something else comes up, and you think that you know particular sessions some user talked about that so you go back to it [Wiki notes].

Participant PT3 discussed how he manages the iteration of design solutions by stating that

> We use a Wiki for our design solutions, we keep the latest design up there.

The rating for practice D3.2 for CS2 and CS4 is "Fully" since a number of methods were used for managing the iteration of design solutions. Participant PT2 utilised the RITE method for testing the updated prototype itself was considered as a record of changes made. Moreover, a pattern library was used to document the iteration of design solutions that could be useful to other projects. Participant PT4 used a shared portal for all the design documents, utilised extreme scenario based design that utilised a central design record that included documentation and direct mapping between design goals, claims and usability testing results and the usability engineer documented meeting notes on usability testing sessions. As for the rating for CS5 it was "Not achieved" since there was no method in place to manage the progress of iterative design except for notes or verbal communication.

Participant PT2 described the utilisation of RITE method by stating that

> It is really just about finding, uncovering problems and fixing them and with this RITE methodology you know the prototype changes like you know 50 times in a course of a week, so you are not even testing the same thing that you started out with, so it is very hard to do any kind of end analysis other than if we are not uncovering any new problems we are good to go.

Participant PT2 discussed how he records the results of iterations of design solutions by stating that

> It [prototypes] is the recording of them [design solutions] but we also have something we have a pattern library so if I make a decision as a designer that this is the component that we are going to use to handle the situation and I know that this is the component that exist in other places in the application or probably would exist in the future then I create a pattern for it [...] and other designers reference it.

Participant PT5 discussed the method used for recording iteration of design solution by stating that

> You know nothing formal to record it, you know just keep notes and talk about it.

Practice D3.3- Use Design Objectives to Control Iteration

The rating for practice D3.3 for CS1, CS3, and CS4 is "Largely" since the prototyping process is managed by setting and monitoring usability measures for particular aspects of the system (CS1, CS4) and in CS3 the results of the design studio drive the development process but there are some instances as discussed in practice D2.2 where iterations are controlled by functionality rather than by design objectives. In CS4 there were also instances when iterations were derived by user needs almost to a fault as illustrated from participant PT4 quote on practice D2.2. The rating for practice D3.3 for CS2 is "Fully" as it can be illustrated from participant PT2 quote on practice D1.2 and D1.3. The rating for CS5 is "No" since design objectives were mostly discarded by the engineering team as declared by participant PT5 comment on practice A2.2.

Participant PT4 discussed how the usability engineers had overdone change based on feedback by stating that

> We were close to the user I would say, but one of the problems we had I think in this project is that we did not do a good a job as we should have in terms of determining what prioritising changes. I mean the user would want something and we would say OK we will do it, like that without considering the impact, without analysing the request and communicating to users the impact of that request, but OK that is something that we got better at towards the end of the project.

The collective rating for Iteration Attribute for CS1 and CS3 is "Largely", "Fully" for CS2 and CS4, and "Not achieved" for CS5.

E.1-Human Centred Leadership Attribute

Human Centred Leadership Attribute is composed of three practices. Practice E1.1- Manage usability programme, E1.2- Systematic improvement of quality in use, and E1.3- Human centred improvement of organisation.

Practice E1.1- Manage Usability Programme

The rating for practice E1.1 for CS1 is "Not achieved" since although efforts were exerted to conduct UCD activities in different projects yet due to time and resource limitations projects had to be prioritised in regards to their need to UCD resources thus UCD processes were managed on some projects only. Participant PT1 described this by stating that

> No there was not [an organisation usability programme] you know I think it was the assumptions was, we do whatever we can, we work on many projects with just resources we have and it was kind of you know because the projects were somewhat prioritised like which one is the most crucial then I kind would just work on them.

The rating for practice E1.1 for CS2 is "Fully" since there is a well established program for managing user centred processes on all projects in the organisation as it was illustrated from description and quotes of practices A2.1, A2.2, C1.1, C1.2, C1.3, C1.4, C2.1, C2.2, C2.3, C3.1, C3.2, C3.3, D1.1, D1.2, D1.3, D2.1, D2.2, D2.3, D3.1, D3.2, and D3.3.

The rating for practice E1.1 for CS3 and CS4 is "Partially". In CS3 efforts were exerted towards managing UCD across all projects via utilisation of the design studio, contextual inquiry, interviews, user colocation, and developing style guides. In CS4 there were some efforts exerted towards managing UCD across all projects yet these efforts are revolved around applying extreme scenario based design. In CS5 there was an absence of an established program for managing user centred processes on all projects in the organisation.

Participant PT2 discussed how UX is integrated into the development life cycle by stating that

> User experience is an ingrained part of our product development process. For example, a user experience assessment is given to every feature at every sprint review as a part of our assessment of whether a product is on track or not and ready to be released or not, it is one of the key decision data points.

> Also, all user research reports are given to everyone at the company.

Participant PT3 discussed efforts exerted to develop style guides by stating that

> A style guide to kind of unite our various interface development efforts towards a common look and feel.

Participant PT5 provided a reason for the lack of a UCD management program by stating that

> The company is very engineering driven.

Practice E1.2- Systematic Improvement of Quality in Use

The rating for practice E1.2 for CS1, CS4, and CS5 is "Not achieved" since there is no efforts exerted to analyse and improve problems with the organisation's processes in regards to usability in order to reduce usability defects. The rating for practice E1.2 for CS2 is "Fully" and for CS3 is "Partially" since some efforts are exerted by all departments to analyse and improve the organisation's processes, however, most of these efforts are "ad-hoc" rather than systematic as expressed by participant PT3.

Participant PT1 expressed an interest to do systematic improvement by stating

> I think that would be great to do.

Participant PT2 described efforts exerted by UX team towards systematic improvement of usability via patterns library by stating that

> We have a pattern library so if I make a decision as a designer that this is the component that we are going to use to handle the situation and I know that this is the component that exist in other places in the application or probably would exist in the future then I create a pattern [...] and other designers reference it.

Participant PT3 expressed efforts exerted by different departments in order to achieve user satisfaction by declaring that

> Everybody [staff] in their own particular discipline is focused on their end users because that is how they measure their success.

Practice E1.3- Human Centred Improvement of Organisation

The rating for practice E1.3 for CS1, CS4, and CS5 is "Not achieved" since there is no procedure in place towards improvement of the organisation's UCD processes. The rating for practice E1.1 for CS2 is "Fully" since not only UCD team has a role in improvement but also product owners and support team. The rating is "Partially" for CS3 since some efforts were exerted by lead UX designer towards making developers more user centred, for example via educating and involving them in conducting user interviews.

Participant PT2 discussed efforts exerted by product managers towards user centred improvement of organisation by stating that

> Overall the product owner is doing that [focused on customer needs] [...] they [product owners] have something called idea exchange which is a website available for our customers where they can input their ideas and vote on each other's ideas so the PMs very easily see what people are sort of screaming for the most [...] there is also the support team who does report on what are the problems that the customers are currently complaining about so they have that input.

Participant PT3 discussed his efforts in educating developers to become more user centred by declaring that

> I just switched teams and when I started looking at their backlog items they were like you know write stored procedure for X, like who this story is for, right, there is always that drift that you kind of have to keep on top of and keep the language user centred.

The collective rating for Human Centred Leadership Attribute for CS1, CS4, and CS5 is "Not achieved", "Fully" for CS2 and "Partially" for CS3.

E.2- Organisational Human Centredness Attribute

Organisational Human Centredness Attribute is composed of two practices. Practice E2.1- Organisational implementation of user centred practices, and E2.2- Acceptance of human centred skills.

Practice E2.1- Organisational Implementation of User Centred Practices

The rating for practice E2.1 for CS1 and CS5 is "No" since although there is implementation of user centred practices and tools and methods that support UCD, yet the implementation of user centred practices and the

utilisation of tools and methods that support UCD is available for projects that has high usability priority rather than for all organisation's projects (CS1) as illustrated in description and participant quote for practice E1.1 while in (CS5) lack of communication among the cross functional team members as a result of geographical separation and reluctance of the engineering team to collaborate with non engineering teams which resulted in lack of iterative refinement of designs and the engineering team interpreted and implemented the designs incorrectly. As a result UCD practices, tools, methods that support UCD and UX team efforts were all non utilised.

The rating for practice E2.1 for CS2 is "Fully" since there exists a central UX team that serves all Scrum teams in the organisation. Organisational implementation was illustrated from description and quotes of practices A2.1, A2.2, C1.1, C1.2, C1.3, C1.4, C2.1, C2.2, C2.3, C3.1, C3.2, C3.3, D1.1, D1.2, D1.3, D2.1, D2.2, D2.3, D3.1, D3.2, and D3.3. The rating for practice E2.1 for CS3 is "Largely" and for CS4 is "Partially" since the organisation is maintaining a focus on user issues and scenario based design is embraced in all projects implemented by the organisation.

Participant PT3 discussed the focus on users by the different departments by stating that

> Everybody [staff] is very customer focused and [...] wants the same thing there is nobody who wants to build bad or unusable software or deliver bad customer experience right so I think a big part of it is not only promoting UX but understanding how other departments and other disciplines approach the same problem right as far as customer experience goes marketing is all about customer experience you know, sales is all about customer experience I mean people [staff] understand how important this stuff is.

Participant PT4 discussed management supportive attitude towards UCD by declaring that

> I think now they [management] have definitely more fully embraced many UCD concepts and practices into the organisation as a whole.

Practice E2.2- Acceptance of Human Centred Skills

The rating for practice E2.2 for CS1, CS4, and CS5 is "Partially" since although there is an acceptance of UCD practitioners yet there is no process

in place for managing the available UCD process. The rating for practice E2.2 for CS2 is "Fully" and "Largely" for CS3 since there is total acceptance of UCD skills and recognition of their pivotal role (PT2, PT3) and a clear system for managing UCD resources (PT2).

Participant PT2 discussed the growth of the UX team as an illustration of organisational acceptance of UX skills by stating that

> Salesforce began as a user centred organisation and continues to be so. When the CEO started the company he hired a vendor to perform usability testing on the very first prototype of our product. It has always been a part of how we operate. I think that you can tell how much any company values something by how much resources they put on it. Our user experience team has grown from 7 to 50 in my five years with the company that is the biggest demonstration of actual value to the company. In our team of passionate UX professionals, everyone is constantly working on ways to improve what we do. It is expected of us to continue to grow, evolve and be an industry leading example of user experience in the enterprise.

Participant PT2 also discussed the ratio between designers and Scrum teams as an indication of organisational support and acceptance of UX skills by declaring that

> We have got an almost a one-to-one ratio designer to Scrum teams, to me means, you know that we can continue to provide this support to our teams at a level that is needed for Scrum to be successful.

Participant PT3 discussed management recognition of the pivotal role played by UX team by declaring that

> Our executive vice president of software engineering [...] is a strong proponent of what we have accomplished with user experience techniques that he puts it on his executive brief so all the executives are familiar now with the work that I do and which my team have done in the past right and how that have impacted.

The collective rating for Organisational Human Centredness Attribute for CS1, CS4, and CS5 is "Partially", "Fully" for CS2 and "Largely" for CS3.

7.2.2 Maturity level Evaluation of Case Studies via UMM-HCS

Table 7.10 shows the results of maturity level evaluation of UMM-HCS process attribute for CS1, CS2, CS3, CS4, and CS5.

Process Attribute	Rating-CS1	Rating-CS2	Rating-CS3	Rating-CS4	Rating-CS5
Problem Recognition	Fully	Fully	Fully	Fully	Partially
Performed Process	Fully	Fully	Fully	Fully	Unknown
Quality in Use Awareness	Fully	Not achieved	Fully	Partially	Partially
User Focus	Fully	Not achieved	Largely	Partially	Partially
User Involvement	Fully	Fully	Fully	Fully	Partially
HF Technology	Fully	Fully	Fully	Largely	Partially
HF Skills	Largely	Fully	Largely	Largely	Partially
Integration	Largely	Fully	Largely	Partially	Not achieved
Improvement	Fully	Fully	Fully	Fully	Not achieved
Iteration	Largely	Fully	Largely	Fully	Not achieved
Human Centred Leadership	No	Fully	Partially	Not achieved	Not achieved
Organisational Human Centredness	Partially	Fully	Largely	Partially	Partially
Maturity Level	Level C	Level A	Level B	Level A	Level Unknown

Table 7.10: All Case Studies-UMM-HCS Process Attribute Maturity Level Evaluation

Table 7.3 showed the desired rating of each attribute in order to achieve the maturity level [68]. On comparing table 7.3 with the results of the five AUCDI case studies that were reported in table 7.10, it revealed that CS1 maturity level was "Level C", CS2 and CS4 maturity level was "Level A" whereas CS3 maturity level was "Level B" and CS5 maturity level was "Unknown".

The reason behind the inability to identify the maturity level of CS5 was related to inability to compute the collective rating of the A.2-Performed Processes Attribute since the rating of its two constituent key practices A2.1 and A2.2 was "Fully" and "Partially" and on checking the advice on the assessment of criteria in UMM-HCS documentation there was no mention of how such values could be combined.

7.3 Revisiting UMM-HCS Study Aims

This study aimed to investigate: first, whether their existed a relationship between the success of AUCDI attempts and usability maturity level. Second, the suitability of Usability Maturity Model-Human Centrdness Scale for utilisation in assessing usability maturity in the context of Agile projects.

Aim 1: Investigating the existence of a relationship between the success of AUCDI attempts and usability maturity level

It was not possible to achieve this aim via UMM-HCS due to problems exhibited with the model's rating system. Table 7.3 shows the desired rating for each attribute in order to achieve the maturity level [68]. It is clear from table 7.3 that UMM-HCS embraces a linear model of upgrading i.e., an organisation cannot be upgraded to maturity level i+1 unless it has largely or fully achieved all practices in maturity level i. The results of the five case studies are shown in the UMM-HCS recording forms shown in tables 7.5, 7.6, 7.7, 7.8, 7.9, and section 7.2.1. These tables reveal that the linear model of upgrading is contradictory to how organisation's perform since an organisation can score high in some of the practices related to a high maturity level even if the organisation is at a low maturity level. An example for that is CS2 whose maturity level was evaluated as "Level A" although it scored as "Fully" in all attributes except B.1-Quality in Use Awareness Attribute and B.2-User Focus Attribute. Attribute B.1 and B.2 are somehow misleading since if the objective of the practice is to assess development teams' knowledge of usability, user consideration, context of use, etc, then other methods besides training could be utilised, for example, knowledge transfer via day to day job experience or knowledge gained from users' feedback or working in different projects.

Aim 2: Investigating whether UMM-HCS is suitable for utilisation in assessing usability maturity in the context of Agile projects

UMM-HCS was not initially developed for Agile software development processes, however, two criteria were set in order to investigate the suitability of UMM-HCS for utilisation in assessing usability maturity in the context of Agile projects. These criteria involves the following:

CR1: The model should not conflict with Agile values and principles

Chapter 2.1, section 2.1.1 and section 2.1.2 discussed the Agile Manifesto and its values and principles. This criteria was set in order to maintain the agility of the development process in case of utilising UMM-HCS.

Practice E1.1, "Manage usability programme. Management of the whole programme of human centred processes on all projects in a department or organisation" could pose a conflict with the Agile value of "Individuals and interactions over processes and tools", however, this practice also works in support of another Agile principle "Continuous attention to technical excellence and good design enhances agility" since the aim of practice E1.1 is to improve the quality of usability across all products.

Moreover, most of UMM-HCS practices are in support of Agile principles. For example, the Agile principle "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software." is supported by practices A2.1, A2.2, C1.1, C1.2, C1.4, D2.1, D2.2, and D3.1. The Agile principle "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage." is supported by practices D2.1, D2.2, and D2.3. The Agile principle "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale." is supported by practice C1.4. The Agile principle "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly." is supported by practices C2.3 and E1.2.

Thus it can be concluded that CR1 is satisfied by UMM-HCS since the model does not conflict with Agile values and principles.

CR2: The model should integrate UCD activities into the overall project plan and throughout the software development life cycle

The reason behind setting this criteria was illustrated in chapter 6, section 6.3. Criteria CR2 was considered as a significant factor for judging the suitability of UMM-HCS for utilisation in assessing usability maturity in the context of Agile projects since the main problem that faces the Agile domain regarding the integration is when to perform the different UCD activities and how to make them more lightweight in order to accommodate the Agile processes iterative and incremental nature.

UMM-HCS is a generic model, i.e., it is not developed for a particular software development life cycle. As a result the model focus is on declaring the important attributes and practices for usability maturity rather than

clarifying the timing or frequency for conducting these practices along the project plan or the phases of the product development life cycle. This is of specific importance in case of Agile development processes since significant part of the integration challenges are related to the iterative, incremental, tight time line nature of Agile development processes as illustrated in chapter 3, section 3.3.

Thus it can be concluded that CR2 is not satisfied by UMM-HCS since the model does not state clear timings and milestones along the Agile development life cycle for the inclusion of UCD activities into the overall project plan along the Agile iterations or sprints.

7.4 UMM-HCS Critique-A Self Assessment Perspective

This section is a critique to UMM-HCS from the perspective of a self assessor who utilised the model on their own to conduct the assessment.

Theoretical Foundations with Respect to Maturation

Chapter 5, figure 5.1 shows that UMM-HCS is derived from a number of models available. Those models were Total systems maturity (TSM) model [155], the Usability Leadership Maturity Model (ULMM) [87], User Centred Design Maturity (UCDM) [70], Crosby [49] and ISO 13407. However, the model documentation does not contain any information on the underpinning rationale behind choosing key practices and placing particular key practices in certain maturity levels.

UMM-HCS author answered a question that was posed in an email to him regarding the rationale behind the maturity level and key practices "Do you have any document that justifies the reason for including the different management practices in the particular levels?"

The Author stated that

> A few old papers are attached. What we did was collate several similar efforts that had been carried out at the same time. The papers list them. Everyone did broadly the same thing - interpreting Crosby. What we had was a clear set of visions of organisational behaviour at each level and had worked out

> the necessary sequence to make it feasible. We then mapped observed good management practices in HCD on to these.

Nevertheless, the utilization of UMM-HCS in the five AUCDI case studies revealed that the placement of some key practices in certain maturity levels was rather ambiguous, for example key practice E2.2- Acceptance of human centred skills, is included as the last practice in the highest maturity level although it should have been placed in a lower maturity level since it is pivotal as a prerequisite for practices A2.1, A2.2, D1.1, D1.2, and D3.3 in lower maturity levels.

As it can be illustrated from chapter 5, figure 5.1 and UMM-HCS author's response that UMMs build upon each other by adopting earlier UMMs. This raises doubts in regards to the theoretical soundness of the UMMs. UMMs also rarely describe or apply scientifically rigourous methods for developing the models. Moreover, the origin of usability maturity models' contents (processes, attributes or practices) is rarely described.

Maturity Paths

UMM-HCS [68] sample recording form provided notes to assessors regarding how to conduct the evaluation and the sequence of utilising the recording form. Note one stated that to "set a maximum level (ceiling) to which the assessment will proceed via an agreement between the assessor and the client organisation in order to speed up the interview especially for organisations which believe to be of relatively low maturity". Another relevant note was note 8 that stated that "repeat the process for the next level until there is no evidence of performance of any practices from that level, or until the ceiling set on assessment is reached" [68].

Thus UMM-HCS linear model of upgrading assumption is overly simplistic that is not in close agreement with the reality of organisations and ignores the complexity of the usability maturity process. It is also apparent from the results of the five case studies indicated in tables, 7.5, 7.6, 7.7, 7.8, 7.9, and 7.2.1 that an organisation can score high in some of the practices related to a high maturity level even if the organisation is at a low maturity level, an example for that is CS2. This finding also contradicts with the advice given in page 20 of UMM-HCS documentation that states that

> It is advisable to start by asking questions about the lowest levels of maturity and move up the scale until it is obvious that the practices are not being achieved. Do not go beyond this level.

If I had followed this advice then it will have led to stopping the evaluation of CS2 at Maturity Level A while this case study clearly signify an organisation that has a clearly high usability maturity level as declared in table 7.6 and section 7.2.1.

Scoring Scheme

The purpose of maturity models assessment criteria/ scoring scheme is to assess the performance of different processes, attributes or practices. Thus it is important for the scoring scheme to be precise and clear. It is important for the corresponding descriptions of assessed criteria to be precise and clear [240].

This was not the case with UMM-HCS model since the model depended on a scoring scheme that evaluated the satisfaction of each practice by one of these values: not achieved, partially achieved, largely achieved and fully achieved as illustrated from table 7.2. The description provided for this scoring scheme was believed to be vague, non detailed and confusing. The rating scheme was also found to be ambiguous, difficult to assess and led to loss of information. The reasons behind that are the wide latitude in the degree of key practice satisfaction. For example, what does "Partially satisfied" imply? Does it imply that the practice is achieved some of the time or achieved all the time with low quality. In addition, the difference between "Partially satisfied" and "Largely satisfied" is also vague.

Attributes, Practices and Advice on the Assessment of Criteria

UMM-HCS documentation [68] provided advice on the assessment of the criteria by advising on the number of and position of the interviewees, number of projects to be assessed, interviews duration, setting the context at the start of the interview, starting, ending and reviewing the interview. Advice was also given on how to elicit the rate given to each practice when in doubt of the exact rating that should be given and how to compute usability maturity levels.

UMM-HCS documentation advice in regards to computing maturity levels was considered as inaccurate, over simplistic and confusing. An example to illustrate this is Note 6 in UMM-HCS documentation, that states that: "If there is some doubt as to how completely a practice is achieved give the benefit of doubt and rate at the higher level of achievement." and Note 10: "Combine the ratings of each attribute. Once again, give the benefit of the doubt and round up if required".

Giving the benefit of doubt means that a particular practice or attribute

can be upgraded from no to partially, partially to largely, or largely to fully. Rounding up can lead to loss of information and misleading results that contradicts the actual purpose of descriptive UMMs (to assess the as-is situation) and prescriptive maturity models (to assess the as-is situation and plan for improvement).

Terminological Chaos

UMM-HCS provides assessors with a description of maturity levels, process attributes and key practices and a sample recording form in order to conduct the maturity level assessment [68]. One of the challenges faced in conducting the self assessment was that using the sample recording form as-is was very difficult since most of the key practices are written either very briefly or incompletely, this was evident when the key practice in the sample recording form was compared with the description of the key practice in UMM-HCS documentation (Appendix I). Thus for the sake of validity, clarity, and completeness it was decided to use the key practice in the sample recording form as guidance, yet depend on the full key practice description. Nevertheless, upon doing this it was discovered that the current wording of some key practices and key practices description has a number of problems. Those problems were redundancy and overlap, terminology chaos, and double barreled as it will be illustrated below. As a results it was decided to prepare a list of questions to pose to interviewees that will cover all aspects related to the key practices and remove all ambiguities and incompleteness.

UMM-HCS documentation utilised some terminology in a form that created contradictions and ambiguities as it will be illustrated below

Human/ User Centred

In the glossary of terms for UMM-HCS [68] it was stated that human/ user centred are defined "as approaches which have as their primary intention or focus the consideration of the interests or needs of the individuals and / or groups which will work with or use the output from a system". Thus this definition suggested that the terms human centred and user centred can be used interchangeably. Nevertheless, in page 19 it was stated that "The quality in use of whole ranges of systems is coordinated and managed for business benefit. The culture of the organisation gains benefit from being user and human centred" which implies that these two terms were different.

An email was sent to the author in order to clarify this situation and this question was posed "Can I interchange the word human centred with user

centred? His reply was "Yes, although there is some claim now that human is broader than user there would not be any different in approach."

User Experience

The use of the term user experience in practices C1.2 and D2.2 is confusing and misleading since it was apparent from the context of the key practices and its description that the author means usability rather than user experience. However, since there exists a tendency in HCI literature to use the term usability and user experience interchangeably and since there is a distinctive difference between the two terms, thus an email was sent to UMM-HCS author to clarify this issue before starting the interviews. The question posed was "In C1.2 do you mean user experience or usability and user experience?" His reply was "User experience. Usability/quality in use is in C1.3.". He also mentioned the following :"What I note is that: in the ten years since the scale was defined some terms have grown in scope: user experience, quality in/of use. This means that assessments made using the meaning of the definitions ten years ago are not strictly equivalent in terms of what was actually done. But, on the other hand understanding of what needs to be done and expectations have advanced. Another way of looking at this is that the tool is robust against changes in practice." Thus to be on the safe side questions were posed that cover both aspects user experience and usability.

Quality in Use

Practices B1.1, B1.2, C1.3, C2.2, C2.3, D1.1, D2.3, and E1.2 used the term quality in use which also caused confusion since quality in use as a term was coined by Nigel Bevan and its scope is much wider than its use in UMM-HCS documentation. Thus an email was sent to the UMM-HCS author in which the following question was posed "When you refer to quality in use, do you mean usability or quality of use with its broader meaning (I am aware of Nigel Bevan work in this area)", he replied stating that "The intent is a broad definition and it is good that this has got bigger since the scale was defined. Nowadays I would say Nigel's big usability, which seems to be similar to quality of use." In another email that was sent to the author with the following question "What do you mean by quality in use techniques?" He replied stating that "Quality in use was pretentious. Usability or HCD techniques/ methods are more understandable". Thus according to the author's clarification HCD was used as an alternative term to quality in use.

Key Practices- Inaccuracy/ lack of preciseness

The assessment criteria should exhibit a high degrees of inter-subjective verifiability, i.e., the corresponding descriptions are concise, precise, and clear in order to discriminate between the different maturity levels as cited in [240].

It was noticed that some of the key practices are not mutually exclusive since it was hard to discriminate between some of the key practices. This is illustrated in key practice E1.3 and E2.1 and also key practice E2.2 and A1.1.

Staff

The word staff was used in 8 different key practices sometimes to refer to all different staff categories and sometimes to refer to human factors staff. In addition, the word human centred staff was also used. This represents both ambiguity and terminology inconsistency. In key practices A1.1, B1.1, B1.2, B1.3, B2.1, B2.2 all they key practices used the word staff to refer to all different staff categories excluding the UX staff since there are other key practices that evaluates the competency of the UX staff, for example, C3.1, C3.2, C3.3. This is problematic since staff has a wide latitude as it involves a lot of parties, for example, developers, documentation staff, QA, product managers and each of those can have a different status in regards to the evaluated key practice. Moreover, in process attribute C.3, Human Factor Skills Attribute, practice C3.2 used the word human centred staff rather than the word staff, while key practice C3.3 used the word staff to refer to human centred staff. However, this did not create as much confusion since the attribute was clearly named human factors skills attribute.

Appropriate

The term appropriate was used in 6 key practices: C2.1, C3.2, C3.3, D1.2, D1.3, and D2.3. This term was also hard to understand and apply. The author was contacted with these concerns via email where four questions were posed regarding this term in practices C2.1, C3.2, C3.3, D1.3, and D2.3. His reply was "I am afraid that the term "appropriate" was used a lot in the document to mean justifiable to the satisfaction of the assessor. What the assessor is to look for is evidence that the organisation had thought about a selection rather than just done what they could do. This does not necessarily mean a highly competence assessor. The skill is in assessing the quality of the procedure used in the selection."

It was difficult to agree with UMM-HCS author's opinion since it was

believed that this requires considerable competence of the assessor especially when there is no guidelines in the supplied documentation on how to judge the appropriateness.

Suitable

The term suitable was used in two key practices: C2.2 and C2.3. This term was also hard to understand and apply. The word suitable is ambiguous and hard to define in the context of C2.2 and C2.3 since no further clarifications were provided in UMM-HCS documentation on how to judge the suitability.

Relevant

The term relevant was also considered to be ambiguous and hard to grasp and UMM-HCS documentation did not provide any further explanation. It was used in only one key practice A2.2.

Practices- Double Barreled

A considerable amount of key practices and key practices descriptions used the words (and/or). This poses a lot of difficulties on the assessor since he has one of two choices: either to pose the key practice as-is and thus risk misleading results in which he does not actually know whether this evaluation is given for the first, second, etc choice or both of them, or to divide the key practice into a number of key practices that are acquired from the key practice description to cover each aspect on its own, yet the downside to this is that he will have to recombine the answers later on since at the end both of them represent the same key practice which could jeopardize the accuracy of his results. In all case double barreled key practices leads to inaccurate and less informative results that hinders and impairs process improvement endeavours. A list of double barreled questions is provided below Practice A1.1, C1.4, C2.1, C2.2, C2.3, C3.1, D1.3, D2.3, E2.1, and E2.2.

7.5 Comparing Maturity Level Analysis Findings Of Nielsen Model and UMM-HCS

Table 7.11 shows a comparison of the maturity level analysis for the five AUCDI case studies via Nielsen and UMM-HCS Models. The investiga-

Case Study	Maturity Level by Nielsen Model	Maturity Level by UMM-HCS Model
Case Study 1	7	C
Case Study 2	8	A
Case Study 3	7-8	B
Case Study 4	8	A
Case Study 5	Unknown	Unknown

Table 7.11: Comparative Maturity Level Analysis for Nielsen and UMM-HCS Models

tion of the existence of a relationship between the success of AUCDI attempts and usability maturity level via Nielsen model gave an indication regarding the existence of a correlation between the success of AUCDI attempts and the AUCDI case study's usability maturity level since successful AUCDI case studies all scored a usability maturity level that ranged from 7-8. In the failed AUCDI attempt, it was not possible to determine its maturity level since there was a wide difference between its practices and any available maturity level and as a result its rated maturity level is "unknown".

The investigation of the existence of a relationship between the success of AUCDI attempts and usability maturity level via UMM-HCS was inconclusive due to reasons related to the model's rating system since UMM-HCS embraces a linear model of upgrading. However, the results of the five case studies gave an indication that the linear model of upgrading is contradictory to how organisation's perform since an organisation can score high in some of the practices related to a high maturity level even if the organisation is at a low maturity level. Thus this rating system led to considerable loss of information as it can be perceived from table 7.10 that reflects that although a number of organisations achieved a number of attributes at higher maturity levels Fully, Largely or Partially yet due to the linear upgrading scheme of UMM-HCS, it was difficult to take the decision to upgrade them to higher maturity levels.

7.6 UMM-HCS Study Challenges and Limitations

This study suffered from a number of challenges that were mostly rooted in the critique raised for the model and that was extensively discussed in section 7.4. In addition, although UMM-HCS provided detailed documentation, yet the description in UMM-HCS documentation for the maturity levels, attributes, and practices was rather difficult to comprehend and required considerable learning effort to understand the model's details and rationale. This also resulted in difficulty in determining the rating for each practice. As a result several emails were exchanged with UMM-HCS model author in order to clarify all ambiguous terminology.

The study also suffered from a number of limitations. Since the aim was to purposefully choose a number of AUCDI case studies in order to reflect a "two tail" design [286] in which cases from both extremes (success and failure) are selected. This endeavour faced publication bias, a phenomena where more positive results are published than negative results [162]. Only one paper in the AUCDI literature [209] reported failure and 46 other papers reported success as illustrated in chapter 3, section 3.2.5.3. It would be very beneficial for the research field if more people would report on failure attempts in order to have failure analysis cumulative knowledge so as to come up with counteractive measures.

Interviews were focused on posing questions regarding published AUCDI case studies in either experience reports or research papers. Although it is preferable not to pose questions regarding events that occurred a long time in the past. Nevertheless, this occured since some of these projects were dated back to 2005 and 2008. This proved to be a bit difficult for both the researcher and the participants. As for the participants they had to exert an effort in remembering some of the details and the researcher had to be alert to redirect the participants to answer the question about the time of this past project rather than the current time.

UMM-HCS documentation advised to conduct interviews with the head of the systems development group and a project manager in order to provide sufficient information regarding the maturity level. Since most of the key practices being evaluated were related to usability issues thus it was preferred to interview lead usability practitioners in order to ensure knowledge of the interviewer on the evaluated key practices. Moreover, all interviewees were the authors of pivotal experience reports or research AUCDI papers and all of them worked as UCD practitioners. The roles for the five UCD practitioners who were interviewed were usability product

manager, usability analyst, usability engineer, lead user experience designer, and team manager for user experience design respectively.

UMM-HCS documentation also advised to assess more than one project. However, the exact number is dependant on the organisation size. This was not possible in this research since it was possible to gain access to a single project per organisation yet there are a number of questions that are related to evaluating the organisation as a whole and these questions were posed as stated by the UMM-HCS documentation on the organisation as a whole.

7.7 Conclusion

Chapter 5 discussed a variety of UMMs and usability assessment models, however, as indicated in 5, section 5.3.1, published UMM related research in the public domain, and on empirical validation in particular is very limited [154]. None of the available UMM models in the public domain was initially created for use in an Agile development process thus they lack details on the timing and the lightweight method for applying the different UCD practices along the Agile development life cycle iterations or sprints. This is of specific importance in case of Agile development processes as significant part of the integration challenges that faces the Agile domain, as illustrated in chapter 3, section 3.3 are related to the timing and lightweight method of performing the different UCD activities in order to accommodate the Agile processes iterative and incremental nature.

The utilisation and application of Nielsen model on five AUCDI case studies for investigating the existence of a relationship between the success of AUCDI attempts and usability maturity level gave an indication of the existence of a correlation between the success of AUCDI attempts and the AUCDI case study's usability maturity level since successful AUCDI case studies all scored a usability maturity level that ranged from 7-8. In the failed AUCDI attempt, it was not possible to determine its maturity level since there was a wide difference between its practices and any available maturity level and as a result its rated maturity level is "unknown". Whereas the investigation via UMM-HCS gave an indication that it was not possible to achieve this aim due to problems exhibited with the model's rating system. UMM-HCS embraces a linear model of upgrading that led to discarding considerable achieved attributes by the five case studies. Moreover, the results of the five AUCDI case studies gave an

indication that the linear model of upgrading is contradictory to how organisation's perform since an organisation can score high in some of the practices related to a high maturity level even if the organisation is at a low maturity level.

Both Nielsen model and UMM-HCS were found to be deficient in their theoretical foundations with respect to maturation, scoring scheme, advice on the assessment of criteria, terminology used and accuracy of some of the key practices. Although both Nielsen model and UMM-HCS do not conflict with Agile values and principles yet both models do not address the specifics, requirements, activities, success factors, and challenges identified within the AUCDI domain and subsequently they do not pinpoint all dimensions and practices involved in the AUCDI process. These issues need to be taken into consideration by any researcher who considers to develop a UMM for the Agile domain. Examples of AUCDI challenges that are not approached by both models are: practices regarding the communication, coordination, and collaboration between UCD practitioners and Agile developers in order to synchronize and complete their work, practices related to design modularization and chunking, UCD practitioner workload, and maintaining communication between the customer and the development team. Another issue that needs to be approached by the developers of UMM for the Agile domain is the features and activities that should be played by some team roles. Examples of those team roles are XP coach, Scrum master, product manager and whose role can impact the integration process.

Thus there is a need for a descriptive multidimensional maturity model for integrating Agile development processes and UCD. Although AUCDI research is growing, nevertheless, the maturation process of Agile development processes and UCD and its constituents has not been directly approached. There is an absence of an AUCDI maturity model that can allow organisations to conduct an analysis of the current state in order to: pinpoint its weaknesses and strengths in deploying Agile processes and UCD, determine whether the organisation is sufficiently mature for AUCDI, and identify the potential difficulties that could develop during the AUCDI process in order to mitigate them beforehand.

7.8 Chapter Summary

This chapter investigated the suitability of Usability Maturity Model-Human Centrdness Scale for utilisation in the Agile domain in order to assess the organisation's UCD capability. It reported on applying UMM-HCS in five case studies that integrated Agile development processes and UCD and utilising the model in assessing their usability maturity level. It offered a critique to UMM-HCS model from a self assessment perspective. The chapter also provided a comparison between the results and structure of Nielsen and UMM-HCS models and made a mapping between their practices. The chapter ended by listing the research challenges faced and the research limitations.

The following chapter will report on building on the knowledge gained from conducting the SLR, applying Nielsen model and UMM-HCS model in developing a maturity model for integrating Agile development processes and UCD.

Chapter 8

Dimensions for Integrating Agile Development Processes and User Centred Design

The objective of this chapter is to propose a set of dimensions that represent fundamental elements that affect the integration of Agile development processes and UCD. These dimensions will act as the foundation for developing a descriptive multidimensional maturity model for integrating Agile development processes and UCD that will be discussed in chapter 9. This chapter argues that AUCDI is dependent on four main dimensions: UCD infrastructure; AUCDI process; people involved in the integration process; and UCD continuous improvement.

8.1 Knowledge Base

A variety of sources were used in order to develop the set of AUCDI dimensions. These sources represented the knowledge base for constructing the AUCDI dimensions and included theoretical sources, literature reviews and empirical sources as illustrated in the following sections.

Theoretical Sources

Since the aim was to provide dimensions for integrating Agile development processes and user centred design that comply/ does not conflict with both Agile values and principles and UCD values and principles thus two theoretical sources were utilised. These sources are: first, the Agile Manifesto so as to ensure that none of the proposed processes or practices for integration conflict with Agile values and principles. Further details on Agile methods are provided in (chapter 2, sections 2.1, 2.1.1, 2.1.2, 2.2). Second, a number of sources for describing UCD principles and activities so as to ensure concrete guidance in regards to integrating UCD into the overall project plan and all phases in the product development life cycle via clear milestones for UCD activities along the software development process. Thus UCD principles and activities discussed in ISO 13407, and UCD Processes from KESSU 2.2 [146] were taken into consideration. Both were discussed in (chapter 2, section 2.6).

Literature Reviews

A number of issues were taken into consideration when the AUCDI dimension were formulated. First, Agile and UCD differences and commonalities since they represent divergence and convergence points that can hinder or enhance the integration. Second, AUCDI integration success factors in order to include them as integration processes or practices in the proposed AUCDI dimensions. Thus two literature reviews were conducted, the first, a SLR that resulted in identifying the differences and commonalities between Agile and UCD, AUCDI challenges, success factors and practices and was reported in (chapter 3 in sections 3.2.7, 3.2.8, and 3.3).

The second literature review focused on usability maturity models and was reported in chapter 5. The results of the UMM literature review revealed the deficiencies of the usability maturity models that are available in the public domain in regards to: the quantity of published research in the public domain in general and on empirical validation in particular and

lack of a UMM that is initially created for use in the context of Agile development processes.

Empirical Sources

Two empirical studies were conducted and their findings, observations and lessons learned provided insights for the proposed AUCDI dimensions. These empirical sources are: first, 14 interviews of AUCDI industrial attempts that is reported in chapter 4 that provided insights in regards to AUCDI difficulties and practices utilised by industrial practitioners to tackle these difficulties. Second, the interview study that utilised Nielsen Capability Maturity Model and UMM-HCS in assessing the usability maturity level of five AUCDI case studies and is reported in chapters 6 and 7. The findings, observations and lessons learned from Nielsen model and UMM-HCS revealed their deficiencies in addressing the specifics, requirements, activities, success factors and challenges identified within the AUCDI domain.

8.2 AUCDI Maturity Model Dimensions

The knowledge base discussed in section 8.1 resulted in the development of four AUCDI maturity dimensions, which will be presented in this section. This chapter argues that successful integration of Agile development processes and user centred design is dependent on four main dimensions: UCD infrastructure, AUCDI process, people and UCD continuous improvement. Throughout this section the word UCD practitioner will be used to refer to the following roles: UCD specialist, usability specialist, interface designer, designer, interaction designer, usability engineer, and user experience designer. These terms are used interchangeably in AUCDI literature.

8.2.1 Dimension 1: UCD Infrastructure

Usability research is situated in an organisational context, requires organisational knowledge, and depend on organisational involvement and can motivate organisational changes. Thus the organisational context could influence how usability work is conducted and how it impacts software development [276]. UCD infrastructure involves a number of organisational elements that need to be available in order to achieve successful

integration of Agile development processes and UCD. The importance of UCD infrastructure is that it signifies a high maturity for AUCDI since in the absence of this infrastructure AUCDI will be dependent on the efforts and interest of a usability champion or development team members in achieving the integration. As a result the integration will occur on per project basis based on the availability of development team skills and interest rather than on an organisational policy that encourages and enforces AUCDI to occur throughout all projects. UCD infrastructure as a component is composed of a number of sub components: funds, staff, tools, methods, management support, training, utilisation of standards, patterns and style guides and colocation of developers and UCD practitioners as it will be illustrated below:

Funds

The first component of UCD infrastructure dimension is related to the allocation and utilisation of dedicated funds for conducting UCD related activities. These funds can be used in conducting early user research, field studies, usability tests, setting usability labs, buying prototyping tools, hiring qualified UCD staff, conducting relevant UCD training programs, setting an open space workplace for UCD practitioners and developers, etc.

Staff

The second component of UCD infrastructure dimension is the presence and utilisation of qualified UCD practitioners. Further details on UCD practitioners qualifications will be provided in section 8.2.3.3.

Tools

Integrating UCD into Agile development processes requires the presence and utilisation of tools. Examples for these tools are usability labs and design, prototyping, and documentation tools, etc.

Methods

Appropriate UCD methods should be used in carrying out different UCD activities. These could include user requirements elicitation methods, for example, questionnaires, interviews, focus groups, observations, thinking aloud, etc. Moreover, these also include usability inspection methods, for example, discount usability testing methods, etc.

Management Awareness and Support

Lindstrom and Malmsten [184] raised the issue of the importance of management support and trust in Agile teams. This importance is attributed

to the lack of requirements or final specification to show up front, prior to the project initiation. This throws an extra responsibility on UCD practitioners to persuade management that a good product will be reached via the Agile methodology and process.

In the context of AUCDI, management should be aware of the importance of UCD and the UCD practitioner's role. They should also understand that UCD must be included in the business strategy and support the inclusion and prioritization of UCD activities in the Agile development process.

Training

Achieving AUCDI requires awareness on both Agile and UCD related aspects. Agile awareness on one hand involves Agile values, principles, and practices. Agile awareness training should be provided to a number of parties: UCD practitioners, customers, and business managers. UCD awareness involves UCD principles and activities. UCD awareness training should be provided to a number of parties: customers, business managers, developers, Agile coaches or Scrum masters and product owners. Finally UCD practitioners should receive awareness training about Agile developer's role and Agile developers should receive awareness training on UCD practitioner's role. This awareness training will help in setting expectations of both parties from each other as well as help them synchronize their efforts and achieve productive communication.

Standards, Patterns and Style Guides

Standards, user interface patterns and style guides help designers learn from others' experience and thus be able to create better designs [241]. Utilisation of standards, patterns and style guides is of special importance to ensure consistency across products and re-usability. In the context of Agile projects this is of special importance since some projects can be developed by small teams and utilization of standards, patterns and style guides help to improve their work. Team formation also varies among different Agile projects and some teams may lack specialized UCD practitioners and as a result UCD work becomes the responsibility of developers. Since developers usually have software engineering education and software development expertise thus they may lack sufficient UCD, usability and user experience knowledge that can be compensated via utilising standards, patterns and style guides.

Colocation of Developers and UCD Practitioners

The SLR in chapter 3, illustrated that colocation of developers and UCD

practitioners in Agile teams offers a number of advantages. These advantages include rapid and instant communication due to the availability of the UCD practitioner to instantly answer developers questions, clarify and address any design issues that emerge, and exchange design in a constant and an ongoing manner. Moreover, the developers are able to influence the design as it progresses and are aware of UCD practitioners' work and can discuss early on any design areas that could have a negative impact on implementation. Thus colocation allows for continuous communication, negotiation, knowledge sharing, instant decision making and improved sense of team.

Nevertheless, the main importance and impact of colocation is that it allows for rapid and continuous communication. Thus in case of the presence of organisational policies, work situations or team commitments that prevent colocation this can be mitigated via introducing a clear process for information sharing, colocating teams for cycle planning meetings, continuous synchronous and asynchronous communication using telecommunication tools, for example, chat, instant messaging, emails, phone, video conferencing, etc.

Table 8.1 shows the traceability of practices and processes Of UCD infrastructure dimension for AUCDI maturity model. It shows how each of the processes and practices that belong to the UCD infrastructure dimension relate to the results of the SLR (chapter 3), and empirical results of industrial interviews (chapter 4), Nielsen Model (chapter 6), and UMM-HCS (chapter 7).

8.2.2 Dimension 2: AUCDI Process

Throughout this section the word *cycle* will be used to refer to both iterations and sprints.

Although UCD infrastructure lays the necessary foundation for integrating Agile development processes and UCD, the success of AUCDI is dependent on a number of other dimensions. The second dimension is the presence of an AUCDI process that takes into consideration the iterative and incremental nature of the Agile development process. This AUCDI process should focus on the planning and implementation of UCD activities and principles into the Agile development life cycle so as to achieve the integration. This involves the following issues: planning for the inclusion of UCD activities in the project plan, providing both developers and UCD

Process/Practice	SLR Section(s)	Industrial Interviews Section(s)	Nielsen Model Practice(s)	UMM-HCS Practice(s)
Funds	3.3.5.2, 3.3.5.6	4.3.1.1, 4.3.1.4	B1.2, B1.3, B1.4	Implicit
Staff	3.3.6	4.3.1.1, 4.3.1.6	C1.1, C1.2	C3.1, C3.2
Tools	3.3.9, 3.3.5.6	4.3.2- Documentation Methods	D1.3	C2.2
Methods	3.3.1.1, 3.3.5.1	D1.1, D1.5	D1.1, D1.5	A2.1, C2.1
Management Awareness and Support	—	4.3.1.1	B1.1	A1.1, C2.2
Training	3.3.4.3, 3.3.8	4.3.1.3, 4.3.1.5	A1.1, A1.2, A1.3, B1.1	B1.1, B1.2, B1.3, B2.1, B2.2
Standards, Patterns and Style Guides	3.3.8, 3.3.10	—	E1.2	E1.1, E1.2
Colocation of Developers and UCD practitioners	3.3.11	—	—	D1.2

Table 8.1: Traceability Of Practices and Processes Of UCD Infrastructure Dimension For AUCDI Maturity Model

practitioners with a road map on their roles and responsibilities, executing UCD activities throughout the Agile cycles and synchronizing the efforts of UCD practitioners and developers, etc.

The proposed AUCDI process is a method independent process that seamlessly integrates AUCDI related activities throughout the Agile development life cycle. The AUCDI process integrates UCD into the overall project plan and all phases in the product development life cycle via clear milestones for UCD activities along the software development process. This is achieved via the inclusion of detailed activities of user requirement gathering, feedback and design evaluation as well as explicating integration work products, work flows, roles, and responsibilities. It is based on the ISO 13407 UCD activities and UCD processes of KESSU 2.2 [147]. However, the AUCDI process extends those founding UCD processes and activities by addressing the specifics, requirements, activities, success factors, and challenges identified within the AUCDI domain.

The AUCDI process is based around a number of processes that are achieved through the implementation of a set of practices that can be perceived as sub processes of a process. Basically these practices describe what needs to be accomplished in order to achieve the process. These processes are: planning the UCD process, user analysis, task analysis, iden-

tification and understanding of user requirements, identification and understanding of UI design requirements, lightweight documentation, synchronization efforts between UCD practitioners and developers, coordination and effective scheduling of UCD practitioners and developers activities, interaction design, user task design, usability evaluation. Each of these processes has a set of subsequent practices.

Processes and their associated practices utilise and produce associated work products that take the form of designs, documents, prototypes, working code, training courses, or individual awareness, etc.

The AUCDI process adopts parallel tracks [203] for coordinating the work of developers and UCD practitioners. Figure 8.1 presents the main AUCDI processes and practices. Further details on the AUCDI process is included in Appendix K, section K.11.

The AUCDI process is divided into early work performed in cycle N-1, then work performed in cycle N, N+1, N+2, etc. as it will be illustrated in this section that details the essential processes and practices that are related to the AUCDI process.

8.2.2.1 Early Work-Cycle N-1

Cycle N-1 is the phase where early work is performed, this phase is often referred to in the AUCDI domain as "upfront design" [35, 39, 59, 83, 89, 89, 113, 131, 132, 136, 165, 193, 193, 203, 209, 272, 283]. Upfront design is a a separate pre-development period that is used for eliciting user requirements, understanding users, user goals and context of use, utilising backlog items to create initial user stories and conducting design up front and ahead of developers in order to achieve a comprehensive system view. It can also be used by the development and quality assurance teams to work on back end features such as selecting the development environment and system platforms or developing features that has a high development cost and low design cost. Some teams use this phase to check the technical feasibility of the software via communication between UCD practitioners and architects. It can also be utilised in acquiring feedback from management and sales department. Upfront design mitigated a number of problems, for example, poor design judgments that result in expensive redesigns or lack of customer value, budget issues, poor prioritization of tasks, expensive and late redesign problems that occur as a result of new or changing requirements; problems in usability; inaccuracy of work estimates [83].

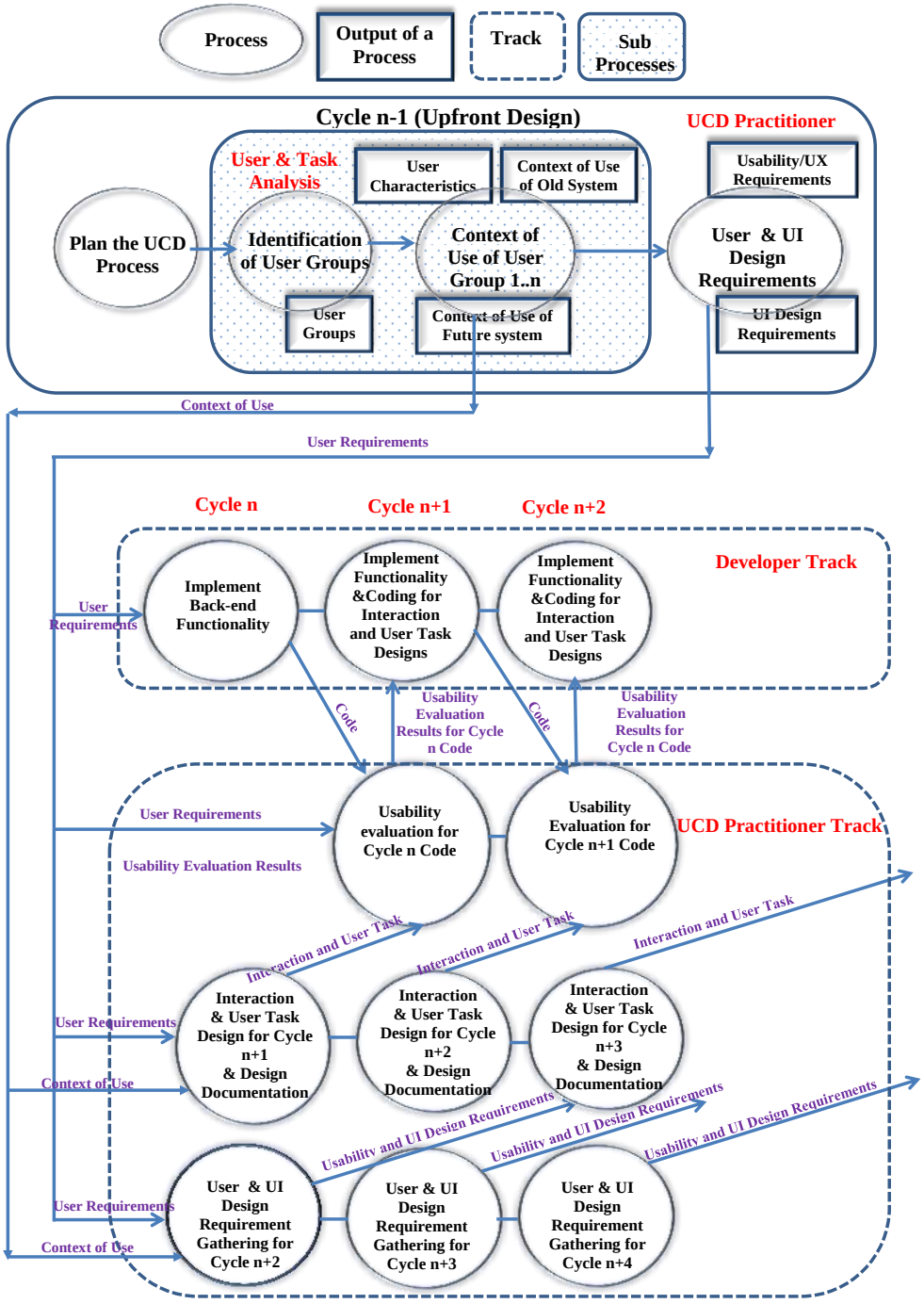

Figure 8.1: AUCDI Process

The early work that is performed in the upfront design phase or cycle N-1 is divided into a number of processes as it will be explained below:

Planning the UCD Process

This process is focused on planning the UCD process and it involves a number of practices as follows:

- **Identifying and planning customer involvement**

  This practice involves the UCD practitioners identifying the most effective method for eliciting customer input at every stage of the Agile development life cycle.

- **Identifying and planning user involvement**

  This practice involves the UCD practitioners identifying the most effective method for eliciting user input at each stage of the Agile development life cycle.

- **Selecting user centred design methods**

  This practice involves the UCD practitioners deciding upon the appropriate UCD methods that will be utilised in the different stages of the Agile development life cycle.

- **Identifying the relevant UCD specialist skills required and planning to provide them**

  This practice involves the management providing the development team with the suitable and necessary skills to perform UCD activities.

Performing User and Task Analysis

This process is focused on user and task analysis and it involves a number of practices as follows:

- **Identification and understanding of user groups**
- **Understanding and specifying the context of use**

  The above two practices involves the identification and understanding of user characteristics, tasks, goals, the organisational, technical, and physical environment of the old product and the technical, organisational and physical environment of the new product.

- **Task Analysis**

  Task analysis involves identifying and understanding the non-functional tasks' attributes, risks, and problems that users face when performing tasks.

Identification and Understanding of User and UI Design Requirements

This process is focused on user and UI design requirements and it involves a number of practices as follows:

- **Identification and understanding of user requirements**

  This practice focuses on the identification and understanding of usability and/or user experience requirements.

- **Identification and understanding of UI design requirements**

  This practice focuses on the identification and understanding of UI design requirements.

Lightweight Documentation

This process focuses on maintaining lightweight documentation for the results of UCD process planning, user analysis and task analysis, user requirements and UI design requirements. Although Agile approaches strive to achieve minimal documentation, however, documentation is crucial for estimation and implementation efforts and for properly integrating Agile and UCD. Documentation is necessary to record the following: firstly, design rationale to justify and record of prior design decisions [193]. Secondly, recording the source of requirements whether they are customers, users, developers, usability experts or usability elicitation guidelines because this can affect the decision of creating new user stories or modifying existing ones since this could have an impact on the workload associated with the respective user stories, and, consequently, on the sprint plan [206]. Thirdly, documenting current designs and their expected delivery date, results of usability tests, progress of late stage design chunks, fixes and recommendations for working versions, and task and user information [272].

It is important to note that documentation is a recurring task that occurs as early as cycle N-1 throughout N, N+1, N+2, etc.

Lightweight documentation can be achieved via a number of lightweight methods, for example, tool support, use cases, wikis, blogs, project web

pages, UI design patterns, personas, sketches, wire frames or prototypes, etc. as it was discussed in details in chapter 3.

8.2.2.2 Cycle Specific Work- Cycle N

Cycle N signifies the starting point for the parallel tracks [203] where the implementation and design are organised as two highly interrelated and equal tracks for effective scheduling and coordination of UCD practitioners and developers activities.

The parallel track involves a cycle 1, where the UCD practitioners work on features that are planned to be implemented in cycle 2. The interface is designed and the testable low fidelity or high fidelity prototype is built. Cycle 1 is also used to rectify and retest design problems that are discovered via usability tests until the designs achieve their design goals. Cycle 1 usually involves developers working on features with high development cost and little UI, whereas the interaction designers work on investigating, creating and verifying next cycles' designs [203]. Parallel tracks are specifically useful in case the features are extremely large and as a result cannot be subjected to design, test for usability, iteration, validation, building, translation and documention in one sprint [76]. The parallel track offers a number of advantages for both interaction designers and developers. UCD practitioners on one hand design for the next iteration thus no time is wasted in creating unusable designs and timely feedback on designs is received from developers. Developers on the other hand maximize time for coding since they do not need to wait for the completion of paper prototypes and usability tests by UCD practitioners [203].

This process involves a number of practices as follows:

- **Synchronizing activities for UCD practitioners and developers**

  UI consistency can be undermined when code is evolved in parallel by independently empowered teams who lack coordination [59]. Brown et al. [34] observed that 92% of the collaborative events between designers and developers were focused on realigning individual work or individual understandings of project and product aims. Furthermore, Budwig et al. [35] declared that UCD practitioners complain that developers' lack of frequent communication of changes creates immense confusion among UCD practitioners and requires the UCD team to exert significant effort to clarify the confusion and deliver on

time. This resulted in added stress on UCD practitioners and created a negative work life balance for them [35].

As a result, the proposed AUCDI process contains a number of synchronization points to allow for close collaboration between UCD practitioners and developers in order to keep the flow of information among all involved parties. Synchronization of UCD practitioners and developers efforts can occur via a number of methods illustrated in chapter 3. The following practices are all used as synchronization points between the UCD practitioners and developers throughout sprint N.

Developers share product vision with UCD team

This practice aims to start sprint N by developers sharing the product vision with UCD team in order for UCD practitioners to understand product vision and cooperate with developers in achieving it.

UCD practitioner shares user information with the development team

Miller [203] declared that the time invested in having a shared understanding and agreement among the entire team on the target users' results in eased collection and utilisation of customer input throughout the development process. This also results in enabling the UCD practitioners to make decisions on design trajectories and feature sets [203]. This practice aims to allow development team to better understand the user, this will be achieved through UCD practitioners sharing the results of user analysis, user requirements, UI design requirements with the development team.

UCD practitioner shares task analysis information with the development team

The aim of this practice is to ensure that developers understand users' needs via understanding the non-functional tasks attributes, problems and risks that users meet when performing tasks.

UCD practitioner shares UCD vision with the development team

The aim of this practice is to ensure that developers understand the design intent and rationale embraced by UCD practitioners. Kollmann [165] stated that the UX vision becomes useless in case of failure to communicate it to the entire team. Ambler [8] declared that the collaboration between usability specialists and software engineers

should be supported via facilitating the communication of design rationale and intent. It is important for UCD practitioners to effectively communicate their understanding of design vision with the development team in order to have a shared design vision. This visibility of design vision offers a number of advantages, for example, creating common understanding of important features to the customer, easier prioritization of features [178], minimizing rework, early illumination of integration [277], ease of making decision in regards to competing concerns between developers and UCD practitioners [178] and more effective development of usable software as a result of active participation of stakeholders that results in improved collaboration [199].

- **Design chunking**

  The aim of this practice is to break design into cycle sized pieces called design chunks that adds incrementally to the overall design and gradually leads to achieving design goals [272]. The incremental nature of Agile processes makes design chunking more critical and challenging [78, 203, 272] as explained in detail in chapter 3, section 3.3. However, design chunking provides the UCD practitioners with the ability to combine different usability investigation methods and to gather more data from less number of users [272].

- **Inclusion of UCD activities (usability/user experience features) in product backlog or user stories**

  The aim of this practice is to ensure the inclusion of UCD activities in product backlog or user stories. The inclusion of usability or user experience related activities in product backlog or user stories is one of the integration challenges. This challenge occurs as a result of developers' focus on accomplishing functionality related features rather than usability or user experience related features [204, 262]. Inclusion of UCD activities in product backlog or user stories usually falls on the shoulders of designers [183], UCD practitioner [204] or usability product owner [35, 262]. The inclusion of UCD activities in product backlog or user stories requires negotiation with developers and Agile coach or Scrum master and product owner as well as awareness of UCD importance by developers, Agile coach or Scrum master and product owner.

- **Prioritization of UCD activities/features in cycles**

  The aim of this practice is to ensure the prioritization of UCD activit-

ies/ features in the development cycles. Prioritization of usability or user experience related activities is one of the integration challenges that were reported as a result of developers' focus on accomplishing functionality features rather than usability or user experience features [204, 262]. Singh [262] reported that the inclusion of usability tasks on the backlog does not guarantee its priortization for implementation in the current sprint [262]. For UCD activities to be prioritized in different cycles this requires both interference from the designers, UCD practitioners or usability product owner as well as awareness of UCD importance by developers, Agile coach or Scrum master and product owner. Prioritization of UCD activities/features in cycles can be achieved via a number of methods that were illustrated in detail in chapter 3.

- **Developers implementing back-end functionality**

  The development and quality assurance teams can utilise sprint N in focusing on back end features such as selecting development environment and system platforms or features that are high on costs of development and low on costs of design.

- **UCD practitioners performing interaction and user task design for cycle N+1**.
- **UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+1**.
- **UCD practitioners gathering user and UI design requirements for cycle N+2**

8.2.2.3 Cycle Specific Work-Cycle N+1

Cycle N+1 is composed of a number of practices as follows:

- **Developers implementing functionality and coding for interaction and user task designs**
- **UCD practitioner continually clearing up any design questions posed by developers**

  This practice represents a synchronization point that is used to ensure that developers understand design

- **UCD practitioners performing usability evaluation for cycle N code**

- **UCD practitioners sharing usability evaluation results for cycle N code with the development team**

  The aim of this practice is to provide a synchronization point between UCD practitioners and developers. Illmensee and Muff [136] reported the advantages of sharing research findings with the development team. These advantages are: allowing the team to discuss usability as well as functionality, giving the team a greater sense of accomplishment, demonstrating a collective concern for quality and user satisfaction to stakeholders. Moreover, Najafi and Toyoshiba [209] stated that the product management and development team were asked to attend user testing sessions in which usability test findings were discussed and designs were reviewed. This was reported to promote sharing of ideas, engage the cross functional team and prepare the development team for the next sprint. Hussain et al. [129] noticed the positive impact created by the attendance of developers to usability tests. He declared that the impact exceeded providing input on design to changing the developers mindsets dramatically to be more user centred.

  Sharing usability testing results and recommendations with development team can be achieved via a number of methods as illustrated in chapter 3.

- **UCD practitioners performing interaction and user task design for cycle N+2**
- **UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+2**
- **UCD practitioners gathering user and UI design requirements for cycle N+3**

8.2.2.4 Cycle Specific-Work N+2

Cycle N+2 is composed of a number of practices as follows:

- **Developers implementing functionality and coding for interaction and user task designs**
- **UCD practitioner continually clearing up any design questions posed by developers**

This practice represents a synchronization point that is used to ensure that developers understand design.

- **UCD practitioners performing usability evaluation for cycle N+1 code**
- **UCD practitioners sharing usability evaluation results for cycle N+1 code with the development team**

  This practice represents a synchronization point between UCD practitioners and developers.
- **UCD practitioners performing interaction and user task design for cycle N+3**
- **UCD practitioners iteratively redoing interaction and user task design for cycle N code according to usability evaluation results acquired in cycle N+1**
- **UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+3**
- **UCD practitioners gathering user and UI design requirements for cycle N+4**

Table 8.2 shows the traceability of practices and processes Of AUCDI process dimension for AUCDI maturity model. It shows how each of the processes and practices that belong to the AUCDI process dimension relate to the results of the SLR (chapter 3), and empirical results of industrial interviews (chapter 4), Nielsen Model (chapter 6), and UMM-HCS (chapter 7).

8.2.3 Dimension 3: People

Another dimension that contributes to AUCDI is the people involved in the process. The AUCDI process involves a number of people: customers, users, developers, UCD practitioners and XP coach in case of utilising XP, Scrum master and product owner in case of utilising Scrum.

8.2.3.1 Customers

Agile approaches require development teams to include customer representatives [14]. In XP teams, the customer represents a fundamental part

Process/Practice	SLR Section(s)	Industrial Interviews Section(s)	Nielsen Model Practice(s)	UMM-HCS Practice(s)
Identifying and planning customer involvement	—	4.3.1.5, 4.3.2	—	—
Identifying and planning user involvement	3.3.5.3	4.3.2-Iteration 0, 4.3.1.4	D1.2, D1.5	A2.1, C1.1
Selecting user centred design methods	3.3.1.1	4.3.2-Iteration 0, Requirement Gathering Methods	D1.2	C2.1
Identifying the relevant UCD specialist skills required and planning to provide them	3.3.6, 3.3.7, 3.3.8	4.3.1.6	C1.1, C1.2	C3.1, C3.2, C3.3
User and task analysis	3.3.1.1	4.3.2-Iteration 0	D1.5	A2.1
Identification and understanding of user and UI design requirements	3.3.1.1	4.3.2-Iteration 0	D1.5, D1.8	A2.1
Lightweight Documentation	3.3.10	4.3.2-Documentation Methods	—	—
Synchronizing activities for UCD practitioners and developers	3.3.4, 3.3.4.5	4.3.1.3	—	—
Design chunking	3.3.2	—	—	—
Inclusion of UCD activities (usability/user experience features) in product backlog or user stories	3.3.3, 3.3.4, 3.3.12	—		—
Prioritisation of UCD activities/features in cycles	3.3.3, 3.3.4, 3.3.12	4.3.1.2, 4.3.1.6	—	—
Developers implementing back-end functionality	3.3.1.1	4.3.2- Iteration 0	—	—
UCD practitioners performing interaction and user task design for cycle N+1	3.3.1.1, 3.3.4.4	4.3.2-Parallel Tracks	—	—
UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+1	3.3.10	4.3.2-Documentation Methods	—	—
UCD practitioners gathering user and UI design requirements for cycle N+2	3.3.4.4	4.3.2-Parallel Tracks	—	—
Developers implementing functionality and coding for interaction and user task designs	3.3.4.4	4.3.2-Parallel Tracks	—	—
UCD practitioner continually clearing up any design questions posed by developers	3.3.4, 3.3.11	4.3.1.3	—	—
UCD practitioners performing usability evaluation for cycle N code	3.3.4.4, 3.3.5	4.3.2-Parallel Tracks	—	—
UCD practitioners sharing usability evaluation results for cycle N code with the development team	3.3.4.6	4.3.1.1	—	—
UCD practitioners iteratively redoing interaction and user task design for cycle N code according to usability evaluation results acquired in cycle N+1	3.3.4.4, 3.3.5.5	4.3.2-Parallel Tracks, Iterating Design	—	—

Table 8.2: Traceability Of Practices and Processes Of AUCDI Process Dimension For AUCDI Maturity Model

of the development team and is expected to be responsible of a set of tasks, for example, generation of requirements and acceptance tests, answering queries of developers, discussing the details of user stories, setting product priorities, providing feedback on iterations and facilitating emergent requirements [15]. The following issues need to be available in order to achieve optimal customer collaboration in teams striving to achieve AUCDI.

- **Identification and Selection of Customers:** This practice aims to identify and select customers since Miller and Sy [204] reported that a weak Agile customer can lead to lack of end user participation and making non informed decisions. Hussain et al. [129] suggested that an experienced on site XP customer can fill in the technical gap between developers and UCD practitioners. XP recommends that the person who acts as a customer in Agile teams should possess a number of features, for example, colocation with the development team; speaking with one voice, being a potential system user, and being collaborative, authorised, representative, knowledgeable and committed [24]. However, this is too idealistic and rarely occurs due to unwillingness of client organisations to spare people to be collocated with the development team, conflicting requirements of different customers; lack of authority of potential system users to make decisions regarding features' identification and prioritization, and lack of understanding of user needs by decision makers. Moreover, Rittenbruch et al. [246] pointed out that the constant exposure of customers to the development process leads to identification of problems related to development and losing focus on user related issues. Sharp et al. [261] stated that since the articulation of the Agile Manifesto efforts has been exerted by XP practitioners to devise methods and practices to deal with the gaps between the reality and the ideal.

  The important issues that are related to customers in order to achieve AUCDI are: awareness, attitude, and continuous involvement.

- **Customer Awareness of Agile and UCD:** This practice aims to ensure that customers are aware of Agile values, principles and practices as well as UCD principles and activities. Understanding Agile on one hand will result in acknowledging the iterative and incremental nature of the Agile development process and what it implies in terms of the time and frequency and form of communication (i.e. face to face). Understanding UCD on the other hand will result in

acknowledging the importance of UCD practitioner's role and the needs and expectations of UCD practitioners from the customer. This will result in understanding and fulfilling the needs and expectations of both the UCD practitioners and developers.

- **Early Involvement:** This practice aims to ensure the early focus on customer needs and thus it is preferable for the customers to be involved from the beginning of the project.
- **Continuous Involvement:** This practice aims to ensure continuous customer involvement since the rapid nature of the Agile development process requires the customer to be continuously involved with the development team in order to answer any questions they may have about the product and also to be involved in usability testing.

8.2.3.2 Users

Software engineering emphasizes the importance of identifying users, understanding their goals and priorities and actively involving them in uncovering requirements [167]. As much as it is important to understand the users it is also important to involve uses effectively in the development process and this entails early and continuous involvement of the right users who correctly represent the larger user population [261]. Blomkvist [23] stated that Agile processes rarely differentiate between customers and users. This resulted in Agile approaches paying little attention to end users and their roles in the development process. However, Sharp et al. [261] declared that XP focuses on users since the on site customer is supposed to be a potential user and to provide feedback on iterations as well as fulfilling many other roles. Nevertheless, there is not so much understanding about how they are selected [246], how many should be involved [246], or who they are although they are responsible for prioritising requirements and choosing what goes in or out of an iteration [261]. Rittenbruch et al. [246] stated that XP does not support design in context. The main reason behind this is that the main communication channel for user requirements are user stories. Users may describe work context and working situations, but they may be focused on system functionality. Thus there is an absence of a means to ensure taking the working context into account. Blomkvist [23] emphasized the need to actively involve users (not just customers) in all development phases. Thus the important practices that are related to users in order to achieve successful Agile and UCD integration are: identification and selection of users, early involvement and

continuous involvement.

- **Identification and Selection of Users:** This practice aims to select the right users who accurately represent the larger user population and involve them in the development process.
- **Early User Involvement:** This practice aims to involve users, who accurately represent the larger user population, in the development process as early as possible.

  Gould and Lewis [95] recommended the direct contact between the design team and potential users via interviews, discussions, and actual observations of users prior to system design rather than learning about users via intermediaries, or through examining user profiles.
- **Continuous User Involvement:** This practice aims to ensure continuous user involvement. Detweiler [59], Ferreira [78], Hodgetts [113] reported that the compressed Agile time scale posed difficulties on usability testing for organising access to the right people at the right time. This is due to the need to plan user involvement, schedule appointments with studies subjects sufficiently in advance and thus may not fit with the Agile development schedule since it may require lead times of weeks. Kollmann [165] stated that the most important factor for including users during the sprints is planning in advance. Continuous quality user involvement can be achieved via user pools, user recruiting firms, remote usability testing as discussed in detail in chapter 3. Larusdottir et al. [172] reported the results of a a survey conducted in Finland on Scrum teams to investigate how usability testing is conducted and some respondents complained about the lack of user commitment and the lack of willingness to take part in usability testing due to being busy doing their own work and lack of desire to be involved in software development.

  Gulliksen et al. [98] recommended the involvement of user representatives throughout the whole project in order to understand the project and become committed to its purpose.

  Continuous user involvement in Agile teams can be achieved by a number of methods as it was illustrated in chapter 3. These methods are: personas, users taking part in writing user stories [246], pairing users with usability designers, low fidelity usability tests, complementing lo-fidelity usability tests with usability acceptance tests, users participation in chartering sessions.

8.2.3.3 UCD Practitioners

Agile teams' structure varies; some teams have a dedicated UCD practitioner and others do not. This can negatively impact the quality of product's usability or user experience. Brown et al. [34] declared that the interface designer place on Agile teams is ill defined. Blomkvist [23] discussed the scarcity of a specialized role in Agile teams with the skills and responsibility to coordinate the work of interaction design. As a result usability and design lies in the hands of the customers or developers and users [23]. Thus customers or users are held responsible to define product features they want, prioritize them, and communicate them to developers [23]. Ferreira et al. [82] stated that the presence of interaction designers on the XP team is essential for enabling good interaction design. The important practices that are related to UCD practitioners in order to achieve successful Agile and UCD integration are: competence, awareness, and attitude.

- **Competence:** Gulliksen et al. [99] stated that the design process participants rarely posse competence, knowledge, interest or special abilities to work in a user centred manner. He pointed out that websters new abridged Oxford edition defines competence as "having the suitable skill for a specific purpose" [99]. He defined competence as "the ability to handle and manage the situation in a specific work context". Competence includes experience and work activity related knowledge, and social skills such as the ability to communicate and cooperate in a group.

  Blomkvist [23] pointed out the deficient focus of Agile processes on competencies essential for the software development projects. Although the role of customers, business people and developers is well described and filled yet the role of usability practitioners and interaction designers is largely overlooked due to lack of awareness among Agile practitioners on usability importance [23]. This lack of professional and skilled UCD staff heavily impacts the design and evaluation of usability in particular and the production of usable products in general [23].

  This practice aims to ensure competence of UCD practitioners. Competence for UCD practitioners involves knowledge and experience of user centred processes, tools and methods [149] in addition to social skills such as communication and cooperation abilities with the rest of the development team.

- **Awareness:** This practice aims to ensure that UCD practitioners are aware of Agile values and principles, what constitutes work for developers, operational challenges and technical challenges and how those challenges impact design [277]. This allows UCD practitioners to set expectations in regards to the time and frequency and form of communication and deliverables with customers, users and developers. It was also reported that when Agile team members perceive the incremental development of user interface as valuable since it resulted in discovering problems early on, uncovered new requirements and helped in solidifying ill defined ones. This results in UX practitioners becoming more accepting to Agile development and more cooperative with developers [178].

  Kollmann [165] reported the results of 10 interviews conducted with UX practitioners to understand whether their understanding of Agile influences their ability to effectively function in an Agile environment. She found out that participants perceive Agile as advantageous since it provide them with more flexibility, ability to iterate the UX design, freedom to choose appropriate UCD practices and methods.

- **Attitude:** Attitude of UCD practitioner involves continuous communication, respect and trust for developers, and maintaining the visibility of UCD team work.

  - **Continuous Communication:** The aim of this practice is to ensure the continuous communication between developers and UI designers in order for designers to answer design questions and receive immediate feedback on design from developers. This continuous communication will help in avoiding delays or bottle necks in the development process. Ferreira et al. [81] reported that the relationship between developers and UI designers is changed as a result of Agile. Ferreira et al. [82] reported that the iterative nature of XP led interaction designers to be continually involved with the product development which impacted the relationship between designers and developers. McInerney and Maurer [193] declared that the Agile development process has UI design to become more of a team effort and created the need for UI designers to be on call to participate in ad hoc discussions. Lee et al. [178] declared that communication between team members is critical and is acknowledged in the Agile Manifesto. However, for teams where developers and

usability engineers work in parallel, collaboration and communication is critical for the software development track and interface design track to remain synchronised.

Ferreira et al. [85] suggested that interactions between UX designers and Agile developers are localized, contingent and purposeful rather than ad hoc. Ferreira et al. [82] declared the importance of having an ongoing and continuous communication between the designers and developers or else delays and bottle necks can occur in the development process.

Albisetti [5] recommended that the UCD practitioner should provide developers with immediate feedback since this allows the developers to bring up issues before they get to the review stage. Continuous communication should be maintained between UCD practitioners and developers where the UCD practitioner is constantly communicating changes and results of usability testing to developers as well as answering any design related questions posed by developers. This allows the UCD practitioners to receive immediate feedback from the developers on problematic issues related to implementing the design as they arise, and the developers are instantly informed with additions or changes to the user interface [82]. Najafi and Toyoshiba [209] reported that the lack of continuous feedback and participation from the UX team in the development process resulted in incorrect interpretation and implementation of designs by the engineering teams.

Ferreira et al. [85] reported that integrating UX design with Agile development is dependent on the recurring efforts of the UX designers and Agile developers to engage together. she reported that work was accomplished via switching back and forth between their own tasks and tasks involving others with explicit and implicit articulation work to coordinate their switching.

Detweiler [59] reported that Agile projects rely heavily on team self governance and operate under highly compressed time scales. This implies that UX managers need to become more actively involved to ensure the regular inclusion of UX activities in team based planning and scheduling. Communication and coordination across teams may also fall more heavily on UX members to ensure compliance to accessibility legislation and user interface style guides. Coatta and Rutter [41] stated that Agile requires

close collaboration between team members and frequent face to face communication, often between stakeholders with significantly different technical vocabularies and backgrounds and as a result one of the biggest challenges facing Agile teams is maintaining clear and effective communication.

- **Respect and Trust for Developers:** The aim of this practice is to optimize communication between developers and UCD practitioners. Ferreira et al. [82] stated that the existence of trust between designers and developers in each other's professional abilities improves the negotiations of conflicting issues and saves time and effort. UCD practitioners' awareness of Agile values, principles and practices as well as what constitutes the role of Agile developers results in increased respect and trust for developers that usually leads to better communication among the Agile team.
- **Maintaining the Visibility of UCD Team Work:** The aim of this practice is to ensure the visibility of UCD team work in order to demonstrate value and promote sustained dialogue. Detweiler [59], Lee et al. [176] emphasized the importance of maintaining visibility of UX team work. They recommend making all "work-in-progress" and deliverables highly visible to the entire team to demonstrate value and promote sustained dialogue. The development team should also be encouraged to frequently check work in progress; post work flows, wire-frames, screen shots, etc., while allowing them to pose questions, challenge, clarify and propose alternatives [59]. UX team visibility can occur via conveying user research results, designs, user testing results throughout the development process. This can be achieved via a number of methods as illustrated in chapter 3.

8.2.3.4 Developers

- **Communication Skills:** The aim of this practice is to ensure that developers have communication skills which is essential for group work, specifically in the domain of AUCDI these skills are needed for the developers to continuously communicate their questions, challenges or requests that are related to design and receive UCD practitioners' feedback related to user research, design and user testing results.

Successful integration of the UX team requires full collaboration and cooperation with cross functional team members [209]. Agile development strongly emphasizes collaboration and interaction between people. The interaction and collaboration between developers and UCD practitioners in Agile teams occur at different times and takes different forms and shapes. This interaction could occur in the form of communication, collaboration or cooperation. Ferreira et al. [85] stated that UX designers and Agile developers interactions were localized, contingent and purposeful. Thus interactions occur for particular reasons, at particular times, driven by their divisional-level commitments to get work done. She pointed out that the integration and coordination of the Agile developers and UX designers work occurred through ongoing negotiations that exposed the work dependencies.

- **Awareness:** The aim of this practice is to ensure that developers have awareness of usability. Jokela and Abrahamsson [152] pointed out the importance of development team's awareness and commitment to usability. Developers need to have an awareness in regards to a number of issues: first, the mechanics of design and how UCD can improve the overall design value and quality [277] or else they may get frustrated due to the time consumed by UCD related activities at early project phases [153]. Second, AUCDI requires mutual awareness between developers and designers; each party should understand what constitutes work for the other party. This mutual awareness will impact informed judgments in regards to work coordination [86]. Thus development team awareness of UCD enables them to understand the importance of UCD work and the amount of effort involved in user research, design, usability testing, etc and thus have realistic expectations from UCD practitioners. This will result in improved communication and collaboration process and increased respect to UCD practitioners.

  This awareness can be enhanced via a number of direct methods(e.g. training) and indirect methods (e.g. the visibility of UCD team work).

- **Attitude:** This practice aims to ensure that developers have the right attitude to ensure the success of the AUCDI process. Attitudes are essential for communication success. In case of people's unwillingness to communicate via listening or sharing their skills and experiences the project will fail regardless of the used tools and methods [99]. Even if the organisation has the necessary UCD infrastructure

and attempts to follow the AUCDI process this cannot be achieved except via possessing the correct attitude that encourages open, clear and continuous communication between developers and UCD practitioners. The developers' attitude is measured via respect and trust for UCD practitioners, and trust for customer.

- **Respect and Trust for UCD Practitioner:** The aim of this practice is to ensure the developers' respect and trust of developers for UCD practitioner, since it is essential for AUCDI success as it allows better communication, cooperation and collaboration. A number of issues contribute to raising this respect and trust, for example, awareness and visibility of UCD team work.
- **Trust Customer:** The aim of this practice is to ensure that developers trust customers. Trust is important as stated in Sharp et al. [261], who reported the results of a focus group of Agile developers attending the Agile development conference in June 2003 where participants reflected on their practical experiences of making customer collaboration work in its different forms. Throughout the sessions participants emphasized the importance of trust between developers and customers as an absolute prerequisite for customer collaboration.

Table 8.3 shows the traceability of practices and processes Of people dimension for AUCDI maturity model. It shows how each the processes and practices that belong to the people dimension relate to the results of the SLR (chapter 3), and empirical results of industrial interviews (chapter 4), Nielsen Model (chapter 6), and UMM-HCS (chapter 7).

8.2.4 Dimension 4: UCD Continuous Improvement

This dimension has a number of practices as follows:

- Presence of a UCD monitoring process across projects.
- Presence of a systematic improvement process for UCD activities, tools, methods, skills and awareness.
- Benchmarking product's usability and/or user experience against competitive products' usability and/or user experience.
- Product decisions emerge from end user and customer studies and are targeted to meet users needs and expectations.

Process/Practice	SLR Section(s)	Industrial Interviews Section(s)	Nielsen Model Practice(s)	UMM-HCS Practice(s)
Identification and selection of customers	3.3.12	4.3.1.5	—	—
Customer Awareness of Agile and UCD	3.3.12	4.3.1.5	—	—
Early Customer Involvement	3.3.12	—	—	—
Continuous Customer Involvement	3.3.12	4.3.1.5	—	—
Identification and Selection of Users	3.3.5.3	4.3.1.4	D1.5	A2.1
Early User Involvement	3.3.5.3	—	D1.5	A2.1, C1.1, C1.3
Continuous User Involvement	4.3.1.1, 3.3.5.3	4.3.1.4	D1.6	C1.1, C1.2, C1.4
UCD Practitioner Competence	3.3.8	—	—	C3.1, C3.2
UCD Practitioner Awareness	3.3.4, 3.3.4.4, 3.3.4.5	4.3.1.3	—	—
UCD Practitioners Continuous Communication	3.3.4, 3.3.4.1	4.3.1.3	—	D1.2, D1.3
UCD Practitioners' Respect and Trust for Developers	3.3.4.3	4.3.1.3	—	—
Maintaining the Visibility of UCD Team Work	3.3.4.1, 3.3.4.2, 3.3.4.6	4.3.1.3, 4.3.2	—	D1.2, D1.3
Developers' Communication Skills	3.3.4	4.3.1.3	—	—
Developers' Awareness	3.3.4.1, 3.3.4.2, 3.3.4.3, 3.3.4.4, 3.3.4.6, 3.3.8	—	A1.1, A1.2, A1.3	B1.1, B1.2, B1.3, B2.1, B2.2
Developers' Respect and Trust for UCD Practitioner	3.3.4.3	4.3.1.3	A1.1	A1.1
Developers' Trust of Customer	3.3.12	4.3.1.5	—	—

Table 8.3: Traceability Of Practices and Processes Of People Dimension For AUCDI Maturity Model

Table 8.4 shows the traceability of practices and processes Of UCD continuous improvement dimension for AUCDI maturity model. It shows how each the processes and practices that belong to the UCD continuous improvement dimension relate to the results of the SLR (chapter 3), and empirical results of industrial interviews (chapter 4), Nielsen Model (chapter 6), and UMM-HCS (chapter 7).

Process/Practice	**SLR Section(s)**	**Industrial Interviews Section(s)**	**Nielsen Model Practice(s)**	**UMM-HCS Practice(s)**
Presence of a UCD monitoring process across projects	—	—	E1.1	E1.1, E2.1
Presence of a systematic improvement process for UCD activities, tools, methods, skills and awareness	—	—	E1.2, E1.3	C2.3, C3.2, E1.2
Benchmarking product's usability and/or user experience against competitive products' usability and/or user experience	—	4.3.1.1	—	—
Product decisions emerge from end user and customer studies and are targeted to meet users needs and expectations	—	4.3.1.1	E1.5, E1.7	—

Table 8.4: Traceability Of Practices and Processes Of UCD Continuous Improvement Dimension For AUCDI Maturity Model

8.3 Chapter Summary

This chapter discussed the proposed dimensions for enhancing the integration of Agile development processes and UCD throughout the Agile development life cycle. These dimensions were based on theoretical sources, literature reviews and empirical sources. The dimensions are: UCD infrastructure, AUCDI process, people, and UCD continuous improvement. The following chapter will discuss the utilisation of these dimensions in the development of a multidimensional descriptive AUCDI maturity model.

Chapter 9

A Maturity Model for Integrating Agile Development Processes and User Centred Design

Chapter 8 discussed the proposed dimensions for enhancing the integration of Agile development processes and UCD throughout the Agile development life cycle. This chapter will utilise these dimensions in the development of a multidimensional descriptive AUCDI maturity model. The chapter details the main phases involved in developing the maturity model and then discusses the AUCDI maturity model and its components.

9.1 Introduction

The assessment of usability/UCD capability via usability maturity models is perceived as a promising research area that can positively impact researchers and practitioners who strive to improve software usability via integrating UCD into their development process. These UMMs aim to assist organisations in conducting a systematic current state analysis that evaluates the organisation's ability to consistently develop products with high and competitive usability level via assessing the organisation's strengths and weaknesses in regards to UCD related aspects and accordingly plan for improvement actions [153].

The conclusion section of chapter 7 discussed the deficiencies of the usability maturity models that are available in the public domain in regards to: the quantity of published research in the public domain in general and on empirical validation in particular and lack of a UMMs that is initially created for use in the context of Agile development process which addresses the timing and lightweight method of performing the different UCD activities in line with the Agile development processes iterative and incremental nature.

Moreover, the conclusion section of chapter 7 discussed the results of utilising Nielsen model and UMM-HCS in five AUCDI case studies. This revealed deficiencies in both models with respect to theoretical foundations of maturation, scoring scheme, advice on the assessment of criteria, terminology used and accuracy of some of the key practices; and in their relevance in addressing the specifics, activities, success factors, and challenges associated with AUCDI.

Although Agile and User Centred Design Integration (AUCDI) research is growing, yet the process of AUCDI maturation and its constituents has not been directly approached. There is an absence of an AUCDI maturity model that can allow organisations to conduct an analysis of the current state in order to: pinpoint its weaknesses and strengths in deploying Agile processes and UCD; determine whether the organisation is sufficiently mature for AUCDI; and identify the potential challenges that could develop during the AUCDI process in order to mitigate them beforehand.

The aim of this chapter is to develop a lightweight, descriptive maturity model for integrating Agile development processes and UCD that contributes to provision of structure of AUCDI efforts via providing organisations with a set of dimensions, processes, and practices that act as a road map

for successful AUCDI. Thus it can be used by organisations for both process definition and process assessment.

The proposed AUCDI maturity model is up to our knowledge the first maturity model to approach the process of AUCDI maturation and its constituents.

9.2 Key Requirements for Agile and User Centred Design Integration Maturity Model

Based on results from chapters (3, 4, 6 and 7), the following key requirements were considered as necessary for developing an AUCDI maturity model.

RQ1: The model should comply / should not conflict with Agile values and principles, in order to maintain the agility of the development process.

RQ2: The model should comply / should not conflict with UCD values and principles. The development process / process model should take into consideration the activities and principles of UCD.

RQ3: The model should provide concrete guidance via integrating UCD into the overall project plan and all phases in the product development life cycle via clear milestones for UCD activities along the software development process. This can be achieved via including detailed activities of user requirement gathering, feedback and design evaluation as well as explicating integration work products, work flows, roles, and responsibilities.

RQ4: The model should assess both capability and performance. The focus of capability assessment falls on the organisation as a whole and it attempts to identify the extent to which UCD is consistently and systematically implemented in the different projects. Whereas, performance assessment focuses on effective implementation of UCD at individual development projects level.

RQ5: The assessment should be lightweight assessment; low overhead so as not to disrupt Agile time lines and low cost so as it would encourage organisations to utilise it. This will be achieved via restricting the assessment to one day.

RQ6: The model should take into consideration Agile and UCD differences as divergence points that hinder the integration.

RQ7: The model should take into consideration Agile and UCD commonalities as convergence points to achieve the integration.

RQ8: The model should detect AUCDI integration challenges in order to pinpoint weaknesses that can hinder the integration.

RQ9: The model should detect and utilise AUCDI integration success factors in order to pinpoint integration strengths that need to be protected.

RQ10: The model's dimensions and base practices should be defined in a clear, non ambiguous and measurable way.

9.3 AUCDI Maturity Model Development Phases

This section discusses the phases that were followed in order to develop the AUCDI maturity model. A number of sources were utilised as the knowledge base to develop AUCDI dimensions and AUCDI maturity model as discussed in chapter 8 in section 8.1 and as illustrated in figure 9.1.

Design science research is a problem solving paradigm that is composed of two pivotal processes; build and evaluate [111, 187]. Design science research aims to construct and evaluate useful innovative artifacts to cope with organisational and human challenges [111]. The design science procedure that was followed for developing the AUCDI maturity model can be illustrated in figure 9.2 that was based on design science research in information system domain [111], however, it was adapted to reflect the research conducted in the the AUCDI Maturity Model domain.

The maturity models' development framework suggested in DeBruin et al. [57], Mettler [201] was adopted. Figure 9.3 summarizes the sequential and iterative phases that can be used in developing maturity models.

9.3.1 Phase 1- Scope

The scoping phase involved taking a number of decisions in regards to the AUCDI maturity model focus, level of analysis, development stakeholders, and dissemination. These decisions helped in distinguishing the AUCDI maturity model from other models, setting boundaries and limit-

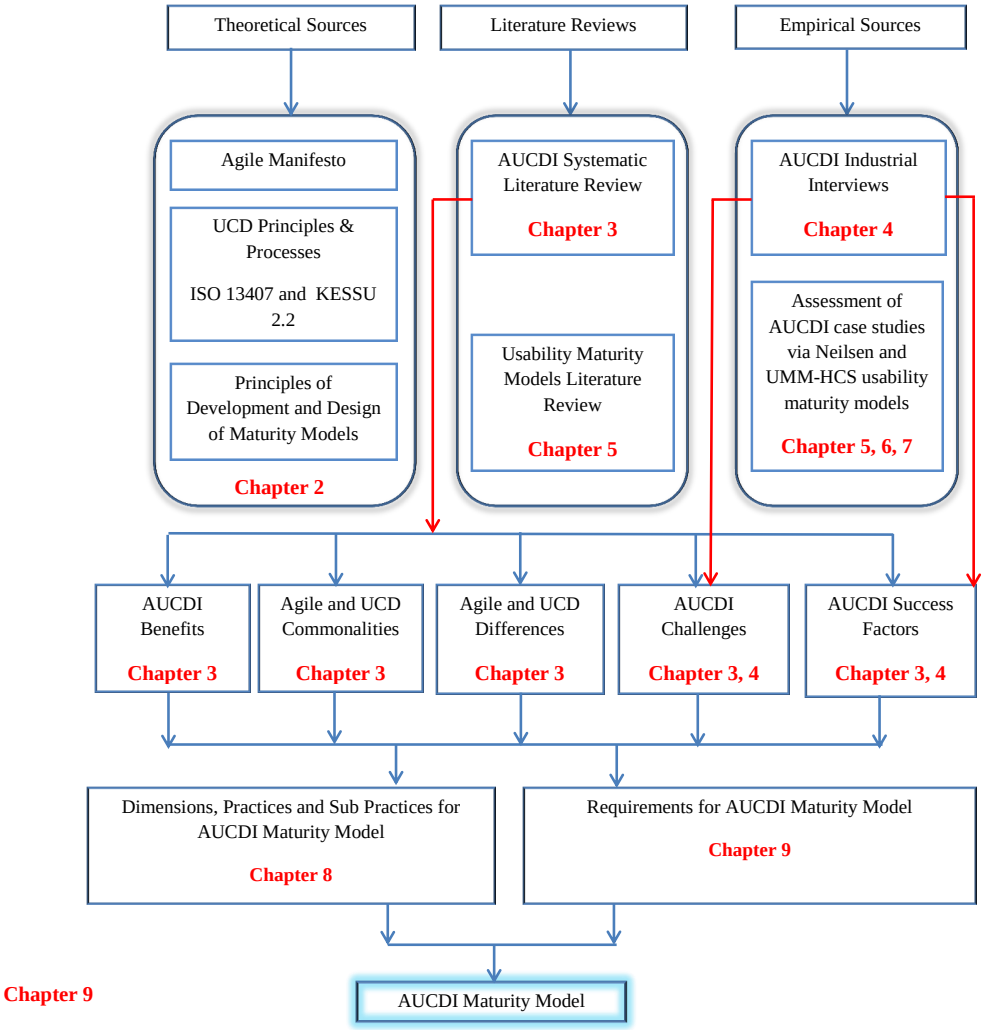

Figure 9.1: AUCDI Maturity Model Sources and Outputs

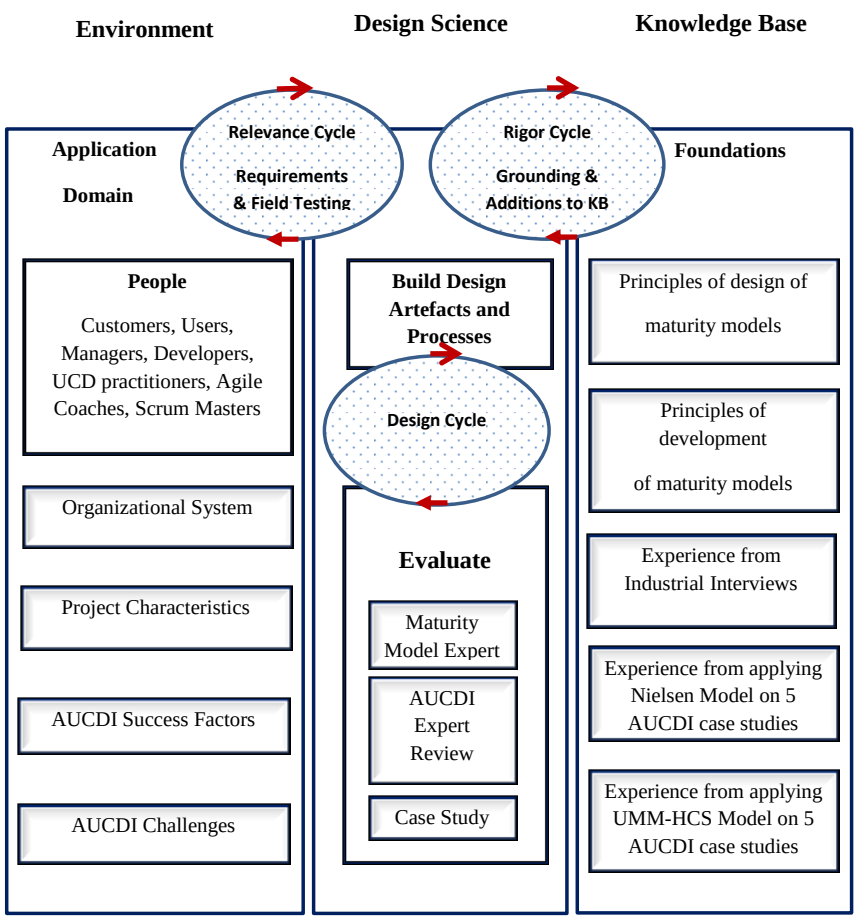

Figure 9.2: Design Science Procedure for Developing AUCDI Maturity Model

Figure 9.3: Development Phases of Maturity Models [57]

ations for the model's application and use, and determining the model's specificity and extensibility.

Table 9.1 shows the decisions related to the scope of the AUCDI maturity model according to decision parameters related to maturity model scoping suggested in DeBruin et al. [57], Mettler [201].

Decision Parameter	**Characteristic**
Focus	Specific Issue (AUCDI)
Level of Analysis	Project, Organisational and Process Factors
Development Stakeholders	Agile and UCD Academics and Practitioners and AUCDI Domain Experts
Agile Method	Scrum and XP
Dissemination	Open

Table 9.1: Design Decisions for AUCDI Maturity Model Scope

Table 9.1 illustrates decisions involved in the scoping phase of the AUCDI maturity model.

Focus

Setting the maturity model focus helps in determining the model breadth. Since maturity models usually start as descriptive models that are used to assess the current situation then evolve into a prescriptive maturity model that provides an improvement road-map once a better understanding of the problem at hand is achieved [57], thus the aim was to develop a descriptive model that can evolve via future work into a prescriptive model that assists improvement road map development from the 'as-is' status to 'to be' status since the only way towards substantial and repeatable improvements is via sound understanding of the current situation. This AUCDI descriptive maturity model will provide organisations with a profound and thorough understanding of AUCDI specifics, activities, success factors and challenges and a diagnostic tool that assesses and describes the current or 'as- is' status.

Level of Analysis

Setting the level of analysis of the maturity model helps in determining the maturity model's depth and operating altitude [57]. The AUCDI maturity model is a multidimensional model that takes into consideration the factors hindering or enhancing AUCDI endeavours. Those factors are UCD infrastructure, people, process and UCD continuous improvement. The model focuses on assessing both capability and performance. The focus of capability assessment falls on the organisation as a whole and it attempts to identify the extent to which UCD is consistently and systematically implemented in the different projects. Whereas, performance assessment focuses on effective implementation of UCD at individual development projects level.

Development Stakeholders

The development stakeholders for the AUCDI maturity model involved industrial practitioners and academics whose work on AUCDI was included in the SLR reported in chapter 3. In addition, the AUCDI maturity model was evaluated by industrial practitioners and academics whose feedback helped in iterating the model into subsequent versions.

Agile Method

The model can be applied to XP and Scrum since the results of the SLR reported on chapter 3 revealed that out of 151 paper on AUCDI (71 included and 80 excluded papers), only two papers discussed AUCDI for Agile methods other than XP and Scrum. Haikara [102] focused on Mobile-D while Krohn et al. [168] focused on FDD thus it was decided to focus on XP and Scrum since they are the main focus of AUCDI industrial practitioners and academics.

Dissemination

The dissemination determines whether the model will be open to specified audience or will be available for exclusive access only. Once the AUCDI maturity model is developed it will be publicly accessible for use by AUCDI academics and practitioners via a website that will be created for that purpose.

9.3.2 Phase 2 - Design

The design phase clarifies the purpose and method of applying the AUCDI maturity model via taking a number of decisions in regards to the AUCDI maturity model's method, driver and level of application, respondents, maturity definition, goal function, design process and design product.

Table 9.2 shows the decisions related to the design of the AUCDI maturity model according to the maturity model decision parameters suggested in [57, 201].

Decision Parameter	**Characteristic**
Driver of Application	Internal Requirement
Respondents	Management, UCD Practitioner, Developers, Agile Coaches or Scrum Master and Product Owner.
Application	Project Level
Maturity Definition	Combination (Process, Object and People focused)
Goal Function	Multidimensional
Design Process	Combination (Theory and Practitioner Driven)
Design Product	Instantiation (Assessment tool)

Table 9.2: Decisions Related to AUCDI Maturity Model Design

Table 9.2 shows that the design phase of the AUCDI maturity model involved the following decisions:

Driver of Application

The AUCDI maturity model driver of application is internal requirements that stem from management, UCD practitioner's or developers' needs to assess the current AUCDI performance in order to pinpoint strengths and weaknesses and plan for improvement.

Respondents

The parties that will be contacted for data collection for conducting AUCDI assessment via the AUCDI maturity model are management, UCD practitioners, developers, and Agile coaches in case of XP projects or Scrum masters and product owners in case of Scrum projects.

Application

The AUCDI maturity model aims to assess both capability and performance; capability assessments identify the extent to which UCD is consistently and systematically implemented in the different projects, while performance assessment focuses on effective implementation of UCD at individual development projects level. Thus elements at both project level and organisation level will be assessed via the AUCDI maturity model.

Maturity Definition

Mettler [201] stated that maturity definition involves defining what entails 'maturity' and it can be process, object, people focused or a combination. The AUCDI maturity definition is a combination of process, object and people focus since all those aspects have an impact on AUCDI. The AUCDI maturity model is process focused since it will assess the practices and activities related to AUCDI. Moreover, it is object focused since it investigates UCD infrastructure, for example, funds, tools, etc. Finally it is people focused since it assesses the skills, awareness and interactions between people involved in the integration. Those people are UCD practitioners, developers, management, customers, and Agile coaches or Scrum masters and product owners.

Goal Function

The goal function of maturity models is concerned with how maturity is represented as either one dimensional, where the entire focus is on a single target measure (i.e., efficiency) or multidimensional (stage gate) where the entire focus is on multiple, sometimes conflicting goals [201]. One dimensional linear stages results in an average maturity stage provided for the entity, thus it cannot adequately represent the maturity of complex domains or provide adequate or sufficient guidance to organisations that aim to improve the as-is situation [57]. Since AUCDI is a complex domain thus it was decided to use a stage gate approach and create the AUCDI maturity model as a multidimensional model. This allowed the sufficient coverage of the specifics, requirements, activities, success factors, and challenges involved in the AUCDI domain. It allowed a chance to provide additional layers of detail in regards to the maturity assessment of discrete dimensions, processes, and practices as well as providing an overall assessment of the organisation. As a result the AUCDI maturity model is capable of providing organisations with a more profound understanding of their strengths and weaknesses and pinpointing areas that need improvement and consequently allowing for more efficient resource allocation. Finally,

stage gate approach allows for drilling down and tailoring the assessment report to the different needs of several audiences.

Design Process

The AUCDI maturity model is a combination of theory and practitioner driven as illustrated from the knowledge base in chapter 8 section 8.1.

The AUCDI maturity model is both theory and practitioner driven since its knowledge base is dependent on a number of sources. Those sources were a SLR that included 71 research papers and industrial reports on AUCDI. In addition the knowledge base included findings from an empirical study of industrial AUCDI attempts and also findings, lessons learned and observations from applying two usability maturity models in five AUCDI case studies. The AUCDI maturity model was evaluated via both academics and industrial practitioners and the feedback gained from this evaluation was utilised in iterating the model into subsequent versions.

Design Product

The quality of the practical usage of the resulting design product is also affected by the design product shape (pure textual description, functioning of the maturity model, instantiation as a software assessment tool) [201].

The AUCDI maturity model will be presented as instantiation rather than textual description. This means that assessors will be provided with all the necessary tools and information to conduct the assessment. Those tools are a multidimensional AUCDI reference model, a performance scale and an assessment procedure. The multidimensional AUCDI reference model will include a set of fundamental elements that affect the integration process and thus should be examined during the assessment. The performance scale will assist organisations in rating the organisational performance in regards to the examined AUCDI elements included in the AUCDI reference model. Finally, the assessment procedure will provide guidance to AUCDI assessors on how to conduct the assessment. This guidance will occur via a maturity recording sheet, maturity levels performance rating, typical quotes and assessment guidelines.

Maturity Stages for AUCDI Model

The design phase also involves decisions in regards to maturity stages. Although AUCDI research started in 2001 and is still growing, nevertheless it has not exploited the research potential on usability maturity models in general and AUCDI maturity models in particular. Until now, the process of becoming more AUCDI mature has not been directly approached. As a

result there is no research available on what constitutes AUCDI maturity and the process of AUCDI maturation. Thus a top down approach was deployed to define maturity stages. Since top down approach works well with relatively naive domains where there is scarce evidence of what represents maturity and involves starting by writing stage definition and then measures are developed to fit the definitions [57].

The maturity stages for AUCDI maturity model are shown in table 9.3.

Stage	**Definition**
Level 0: Not Possible	AUCDI is discouraged. There is a general unwillingness to integrate UCD into the Agile development process. Management and development teams do not seem to value or understand UCD.
Level 1: Possible	AUCDI is not discouraged. There exists a general willingness to perform it. One or few development team members understand the value and meaning of UCD and the benefits of AUCDI.
Level 2: Encouraged	Organisational culture encourages AUCDI. Value, benefits and meaning of UCD is recognised. AUCDI is achieved in some projects.
Level 3: Enabled/Practiced	AUCDI is practiced. UCD methods, tools, workspace and qualified staff exist to enable the integration activities. Management supports and promotes the integration of Agile and UCD.
Level 4: Managed	Employees are expected to perform AUCDI. Training is available. AUCDI activities are part of the software development life cycle. UCD methods, tools, workspace, cycles and qualified staff for supporting AUCDI activities are available. Team Leaders (Agile coaches or Scrum masters and product owners) exhibits awareness and commitment to UCD and provides a strategy for achieving Agile UCD integration throughout all development projects. Development team exhibits awareness and commitment to AUCDI. Customer(s) exhibits awareness and commitment to AUCDI.
Level 5: Continuous Improvement	AUCDI processes are reviewed for assessing the status-quo and plan for improvement. UCD methods, tools, guidelines, workspace and qualified staff are widely accepted and regularly monitored and continuously improved.

Table 9.3: Maturity Stages for AUCDI Maturity Model

9.3.3 Phase 3 - Populate

AUCDI Maturity Model Domain Components

The populate phase focuses on determining the model content via focusing on what needs to be measured and how to measure it via identifying domain components and sub components. A domain component is a major, independent aspect that is significant to a particular domain maturity e.g. critical success factors, barriers to entry [57].

Figure 9.4 illustrates concept diagram identified for the AUCDI Maturity Model. Figure 9.4 shows that the AUCDI components are: UCD in-

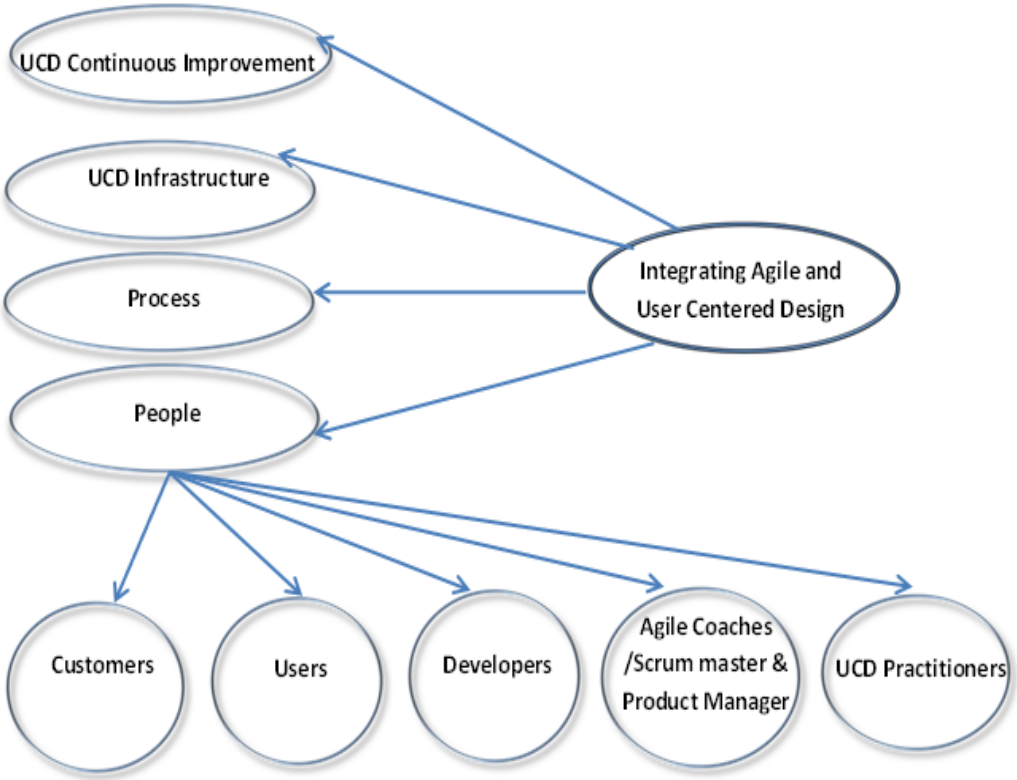

Figure 9.4: Concept Diagram for AUCDI Maturity Model

frastructure, AUCDI process, people (customers, users, developers, Agile coaches/Scrum master, product owners, and UCD practitioners), and UCD continuous improvement. The sources, need for those components and what they entail were discussed in detail in chapter 8.

AUCDI Maturity Model Domain Sub Components

DeBruin et al. [57] stated that domain sub components assist in the development of assessment questions used in the maturity questionnaire, enable richer analysis of maturity results, represents specific capability areas that enable targeted maturity level improvements, and improve the ability to present maturity results in order to meet the needs of target audience. The goal is to attain domain components and sub components that are collectively exhaustive and mutually exclusive [57]. Figure 9.5 represents domain components/dimensions and sub components identified for the AUCDI maturity model. Those components/ dimensions and sub components were discussed in detail in chapter 8.

9.3.4 Phase 4 - Test

After the population of the maturity model, the model needs to be tested. Testing should focus on two aspects: the model's construct and the model instruments [57]. Content validity is assessed as to how domain representation is complete. The literature review extent and breadth of the covered domain provides a measure of content validity [57]. Content validity was addressed via conducting a systematic literature review on the AUCDI domain, that was reported in chapter 3, in order to ensure that the model's theoretical basis is sound.

Face validity is assessed by the achievement of good translations of the constructs. The maturity model is considered accurate and complete with respect to the identified scope of the model. Populating the model via complementary methods assist in achieving face validity [57]. Face validity was addressed in the AUCDI maturity model via a set of complementary methods that involved theoretical sources, literature reviews and empirical sources and was reported in chapter 8, section 8.1.

Moreover, March and Smith [187] points out that maturity model constructs should be tested for simplicity, completeness, ease of use, understandability, operationality, efficiency and impact on the environment and users. Whereas the model instruments need to be tested for validity to ensure they measure what it was intended to measure and reliability to verify that obtained results are repeatable and accurate [187].

Thus an evaluation form was designed in order to evaluate the various aspects related to the model construct and instruments (maturity recording

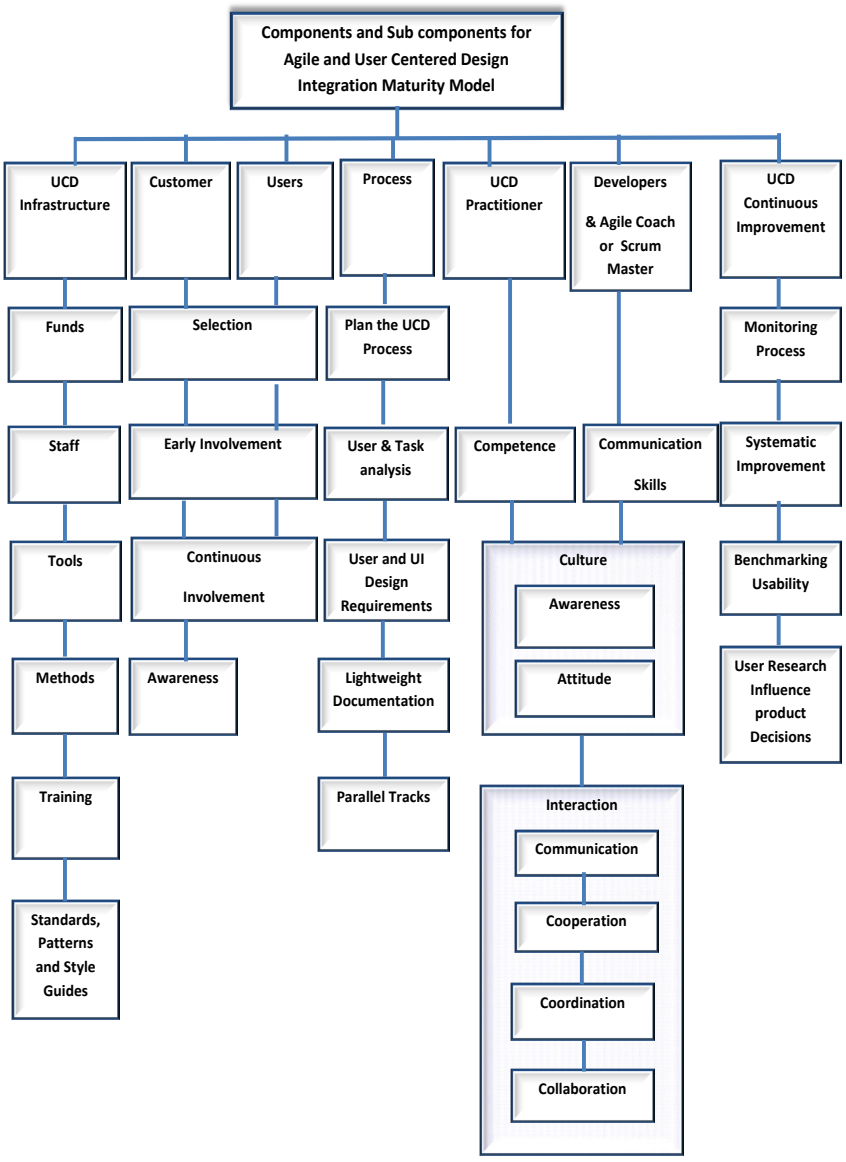

Figure 9.5: Domain Components and Sub Components for the AUCDI Maturity Model

sheet, maturity levels performance rating, typical quotes, and assessment guidelines). This evaluation form will be discussed in details in Figure 9.6.

Helgesson et al. [106] conducted a systematic mapping study on maturity models evaluation and assessment. Helgesson et al. [106] then proposed a maturity model evaluation framework [106]. These maturity model evaluation types are described as follows:

- **Author Evaluation:** The first evaluation type for maturity models is conducted via its authors, who depend on their knowledge of relevant maturity models in general and the maturity model specific domain in particular. This type of evaluation occurs via the authors' evaluating the maturity model's processes for its intended use or by comparing it with other similar maturity models. This method is considered to be cost and time effective since it only involves the model creators who have sufficient knowledge about the model and can dedicate time to elaborate the different model aspects [106].
- **Domain Expert Evaluation:** The second evaluation type occurs via domain experts. This type of evaluation occurs by involving practitioners, who are the experts on the type of process that the maturity model intends to improve, but who have not been involved in the actual development of the maturity model. This evaluation is usually performed via surveys, interviews, or simulated assignments. The success of this method is dependent on the cooperation of the domain experts with the maturity model authors throughout the evaluation process(i.e, providing timely, clear, and detailed feedback,etc.) [106].
- **Practical Setting Evaluation:** The third evaluation type involves using the maturity model in practical settings; for example, for descriptive maturity models this implies utilising the maturity model in diagnostic activities. In case of prescriptive maturity models it implies subjecting the model to diagnostic and process improvement activities. This method is considered to be the most costly, however, the result of the evaluation can be used to analyse and improve both the examined process and the maturity model [106].

The feedback from the three evaluation methods is used for iterative maturity model development and they are usually conducted in the same order discussed above (author evaluation, domain expert evaluation, practical setting evaluation) [106].

9.3.4.1 Evaluation Phase One:Author Evaluation for AUCDI Maturity Model

The first type of evaluation was an author evaluation that was conducted via PhD supervisors, Professor Richard Paige and Dr. Paul Cairns. The evaluation process involved examining the model development phases and the AUCDI model as a product of design then the results of this evaluation led to the evolution of the model into subsequent versions as a result of author evaluation. Appendix K, section K.17 shows the changes made to the model as a result of the author's review.

9.3.4.2 Evaluation Phase Two:Expert Evaluation for AUCDI Maturity Model

The second type of evaluation that was conducted for the AUCDI maturity model involved a set of domain experts evaluating the maturity model. The domain expert evaluation process involved a number of steps: choosing domain experts, inviting them to take place in the evaluation, evaluating the AUCDI maturity model and the evolution of the model into subsequent versions as a result of domain experts feedback.

Choosing Domain Experts

The first step of AUCDI expert evaluation involved choosing an expert panel. This step was highly facilitated by the results of the SLR, which gave an overview of the key domain experts whether from industry or from academia. The selection of the AUCDI domain expert panel occurred via preparing a preliminary list of potential candidates who are experts in the domain of integrating Agile and UCD. This preliminary list was assessed and approved by PhD supervisors. The list of selected AUCDI domain experts is shown in table 9.4.

No.	Name	Relevant Publications
1	Mona Singh	[262]
2	Jim Ungar	[277, 278]
3	Jason Chong Lee	[174, 176, 177, 178]

Table 9.4: AUCDI Domain Experts

Inviting AUCDI Domain Experts

The second step of AUCDI expert evaluation involved inviting AUCDI domain experts via an email that is shown in Appendix K, section K.14. Later an email was sent with two attachments: an informed consent form shown in Appendix K, section K.15 and the AUCDI maturity model documentation.

Figure 9.6 shows the AUCDI domain expert evaluation form that was designed in order to be used to conduct the evaluation

The AUCDI domain experts evaluated the following aspects of the AUCDI maturity model:

- The maturity levels for sufficiency.
- The maturity level descriptions for accuracy and clarity of discrimination between levels.
- The domain components and sub components, that were translated into processes and practices, were evaluated for relevance to AUCDI, comprehensiveness to depict the various aspects relevant to AUCDI, and mutual exclusion.
- The processes and practices were evaluated for correct assignment to their respective maturity level.
- The scoring scheme was evaluated for ease of use and practicality.
- The maturity model documentation was evaluated for understandability and ease of use.
- The maturity model assessment guidelines were evaluated for understandability and ease of use.
- The maturity model was evaluated for usefulness of providing organisations with the following:
    1. A set of dimensions, processes, and practices that act as a road map for successful AUCDI.
    2. An understanding of AUCDI roles, timing, responsibilities, success factors, and challenges.
    3. A diagnostic tool to assess AUCDI challenges that could develop during the AUCDI process.

THE UNIVERSITY of York

Computer Science Department

Agile and User Centered Design integration Maturity Model (AUCDIMM) – AUCDI Domain Expert Evaluation Forms

Expert Information					
Date					
Name (Optional)					
Organization/Institute					
Position					
Email					
Criteria	**Strongly Disagree**	**Slightly Disagree**	**Neither Disagree Nor Agree**	**Slightly Agree**	**Strongly Agree**
Maturity Levels					
The maturity levels are sufficient to represent, all maturation stages of AUCDI (Sufficiency)					
There is no overlap detected between descriptions of maturity levels (Accuracy)					
Processes and Practices					
The processes and practices are relevant to AUCDI (Relevance)					
Processes and practices cover all aspects impacting/ involved in AUCDI (Comprehensiveness)					
Processes and practices are clearly distinct (Mutual Exclusion)					
Processes and practices are correctly assigned to their respective maturity level (Accuracy)					
AUCDI Maturity Model					
Understandability					
The maturity levels are understandable					
The assessment guidelines are understandable					
The documentation is understandable					
Ease of Use					
The scoring scheme is easy to use					
The assessment guidelines are easy to use					
The documentation is easy to use					
Usefulness					
The maturity model is useful for providing organisations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI.					
The maturity model is useful for providing organisations with an understanding of AUCDI roles, timing, responsibilities, success factors, and challenges					
The maturity model is useful for providing organisations with a diagnostic tool to assess AUCDI challenges that could develop during the AUCDI process					
The maturity model is useful for providing organisations with a diagnostic tool to assess AUCDI success factors					

Q1. Would you add any maturity levels? If so please explain what and why?
Q2. Would you update the maturity level description? If so please explain what and why?
Q3. Would you add any processes or practices? If so please explain what and why?
Q4. Would you remove any of the processes or practices? If so please explain what and why?
Q5. Would you redefine/update any of the processes or practices? If so please explain what and why?
Q6. Would you suggest any updates or improvements related to the scoring scheme? If so please explain what and why?
Q7. Would you suggest any updates or improvement related to the assessment guidelines? If so please explain what and why?
Q8. Would you like to elaborate on any of your answers?
Q9. Could the model be made more useful? How?
Q10. Could the model be made more practical? How?
Q11. Are you interested in further usage of the model in a case study evaluation?

Figure 9.6: Agile User Centred Design Integration Domain Expert Evaluation Forms

The members of the AUCDI domain expert panel were encouraged to elaborate on their answers and to suggest any justified updates or improvements related to maturity levels, processes, practices, scoring scheme, assessment guidelines. The result of the AUCDI maturity model expert evaluation led to the evolution of the original AUCDI maturity model into a number of subsequent versions. The changes to the model's maturity levels, key practices, scoring scheme, and assessment guidelines and the reasons behind these changes were recorded and analysed. The maturity model version, date of change, expert reviewer details, and changes made to the model and its underlying reasons were also recorded.

Appendix K, section K.17 shows the changes to the model's maturity levels, key practices, scoring scheme, and assessment guidelines as a result of the expert evaluators review.

9.4 AUCDI Maturity Model Components

The AUCDI maturity model is composed of three components: first, a multidimensional reference model that has a set of fundamental elements that affect AUCDI and thus should be be reflected in the model and examined in an assessment. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. Third, an assessment procedure to provide practical guidance for performing the assessment. Further details on these components is discussed in the following sections:

9.4.1 Multidimensional AUCDI Reference Model

The first component of the AUCDI maturity model is the multidimensional AUCDI reference model. This reference model is composed of a set of dimensions that represent fundamental elements that affect the integration of Agile development processes and UCD and thus should be examined in an assessment. These elements are included in the AUCDI reference model. The results of the assessment can help organisations assess their current status and all the factors that impact AUCDI process and pinpoint weaknesses and strengths in order to pinpoint improvement areas. Full details on these dimensions are included in chapter 8. These elements are as follows:

- **Dimension 1: UCD Infrastructure**

  UCD infrastructure involves a number of organisational elements that need to be available in order to achieve successful integration of Agile development processes and UCD. UCD infrastructure as a component is composed of a number of sub components: funds, staff, tools, methods, management awareness and support, training, standards, patterns and style guides and colocation of developers and UCD practitioners in Agile teams. Further details on the UCD Infrastructure dimension is included in chapter 8.

- **Dimension 2: AUCDI Process**

  This dimension involves the examination of different UCD activities to identify the extent to which they are effectively performed. The basic assessment scope is to examine the AUCDI processes and practices that are performed in individual projects.

  Examples of processes are: planning the UCD process, user analysis, task analysis, identification and understanding of user requirements, identification and understanding of UI design requirements, lightweight documentation, synchronization efforts between UCD practitioners and developers, coordination and effective scheduling of UCD practitioners and developers activities, interaction design, user task design, usability evaluation. Each of these processes includes a set of subsequent practices. Further details on the AUCDI process dimension is included in chapter 8 section K.9.1.2.

- **Dimension 3: People**

  This dimension represents all the people who are involved in the AUCDI process and have an impact on its success or failure. Those people are customers, users, UCD practitioners, developers, Agile coaches or Scrum masters and product owners.

  - **Customer:** The items that are examined in the customer assessment are: the process of identification and selection of customers, Agile and UCD awareness, early customer involvement and continuous customer involvement.

  - **User:** The items that are examined in the user assessment are: the process of identification and selection of users, early user involvement and continuous user involvement.

  - **UCD Practitioner:** The items that are examined in the assess-

ment related to UCD practitioners are: competence, Agile awareness related to Agile values, principles and practices and Agile developer role, attitude that examines UCD practitioner's continuous communication, respect and trust for developers, and maintaining visibility of UCD team work throughout the Agile development process.

  - **Developers and Agile Coach or Scrum Master and Product Owners:** The items that are examined in the developers and Agile coach or Scrum master and product owner assessment are: communication skills, UCD awareness, attitude that examines respect and trust for UCD practitioner, and trust for customer.

  Further details on the people dimension is included in chapter 8.

- **Dimension 4: UCD Continuous Improvement**

  The items that are examined in the assessment related to UCD continuous improvement are: presence of a UCD monitoring process across projects, presence of a UCD systematic improvement process for UCD activities, tools, methods, workspace, skills and awareness, benchmarking product's usability and/or user experience against competitive products' usability and/or user experience and product decisions emerge from end user and customer studies and are targeted to meet users' needs and expectations.

Further details on the UCD continuous improvement dimension is included in chapter 8. Appendix K, section K.11 represents a recording sheet that can be used by assessors to rate the elements included in the AUCDI multidimensional reference model.

9.4.2 Performance Scale

The second component of the AUCDI maturity model is the performance scale (scoring scheme). This scale helps the assessors to rate organisational performance in regards to the examined AUCDI elements included in the AUCDI reference model. The closer the organisation achieves the AUCDI reference model requirements, the higher its ratings. The scoring scheme chosen aimed to make the scoring for the degree of practice satisfaction accurate and informative. Thus a number of scoring schemes were considered as follows:

- Yes/No (Used by Nielsen Model)
- None, Partially, Largely, and Fully achieved (Used by UMM-HCS Model)
- 1-5 Scale

The observations and lessons learned from the evaluation of Nielsen Capability Maturity Model [210, 215] that were reported in chapter 6, UMM-HCS maturity model that were reported in chapter 7 and observations made in Visconti and Cook [280] on version one of a system documentation process maturity model led to a different decision. Yes/No scoring scheme typically attempt to determine whether certain tasks or activities are performed or whether certain circumstances or conditions exist. Although Yes / No as a scoring scheme was simple, straightforward, and easy to utilise, however, there is a considerable latitude in each answer for the degree of satisfaction of each specific practice. This latitude needs to be captured or else a lack of information would occur. Another scoring scheme was also considered (None, Partially, Largely and Fully achieved) that was utilised in UMM-HCS maturity model [68], however, this rating scheme was also found to be ambiguous, difficult to assess and led to loss of information. The reasons behind that are the wide latitude in the degree of key practice satisfaction.

A suggestion to avoid such ambiguities was given in Visconti and Cook [280], this suggestion was implemented in version two of system documentation process maturity model [280] where a five point scale was used that included the following ratings: very high (>4.5), high (3.5-4.5), medium (2.5-3.5), and low (1.5-2.5), very low(>0 and <1.5). This scoring scheme simplifies the computation of the degree of satisfaction of practices and provides easily interpreted names for each of the five degrees of satisfaction and the numerical score provides a finer grain measure.

Thus in order to eliminate those ambiguities it was decided to use the suggestion implemented in version two of system documentation process maturity model [280]. Thus the numerical score and the degree of satisfaction were both used. This scoring scheme can distinguish between barely reaching a particular degree, being in the middle or being at the high end. A "No" (0) option was also included to cover for the absence of a particular practice and a "don't know" option in case of respondent's inability to answer one or more of the questions included in maturity questionnaire due to lack of knowledge or non applicability of the question to the respondent situation. Thus 6 point scale and the "don't know" option were used. This

new chosen scoring scheme allowed the assessment of practices in a much more accurate and informative manner.

The AUCDI performance scale is included in Appendix K, section K.10 to differentiate between the different maturity levels that were included in Appendix K, section K.8.

9.4.3 Assessment Procedure

The third component of the AUCDI maturity model is the assessment procedure. The AUCDI assessment procedure provides guidance to AUCDI assessors in their endeavour to assess the organisation's and project's capability to integrate Agile and UCD. The assessment procedure is composed of a maturity recording sheet, maturity levels performance rating, typical quotes, and assessment guidelines as it will be illustrated in the following sub sections:

9.4.3.1 Maturity Recording Sheet

The maturity recording sheet is used to provide assessors with a template in which they can record their assessment of the different AUCDI processes and practices. The AUCDI maturity model recording sheet is included in Appendix K, section K.11.

9.4.3.2 Maturity Levels Performance Rating

The AUCDI maturity levels with their corresponding performance rating are included in Appendix K, section K.10. This can be used later by the assessors to compare the recorded scoring from the maturity recording sheet with the performance rating of the AUCDI maturity levels in order to determine their organisational AUCDI maturity level.

9.4.3.3 Typical Quotes

A number of typical quotes were put together that signify each AUCDI maturity level. These quotes can provide assessors with a benchmark to compare their maturity level with. The AUCDIMM assessor should use

these quotes in order to ensure that he has correctly assessed the organisational AUCDI maturity level. The inclusion of typical quotes as a guideline for assessors was used in Crosby [49], Earthy [68] in Quality Maturity Grid and UMM-HCS respectively. Appendix K, section K.12 lists a set of typical quotes per maturity level.

9.4.3.4 Assessment Guidelines

A set of guidelines for performing the assessment were put together in order to provide assessors by a clear road map for conducting the assessment. Those assessment guidelines are included in Appendix K, section K.13.

9.4.3.5 Assessment Report

The results of the assessment is utilised in generating an assessment report. This report includes an executive summary with the maturity level and an AUCDI maturity profile. The AUCDI maturity profile indicates the degree of satisfaction of each key practice and whether it is unsatisfied, missing or needs improvement.

9.4.3.6 Constructs

A construct forms the domain vocabulary. March and Smith [187] elaborated the meaning of constructs as follows: "Constructs constitute a conceptualization used to describe problems within the domain and to specify their solutions. They form the specialized language and shared knowledge of a discipline. They define the terms used when describing and thinking about tasks". Thus a set of constructs were prepared related to the domain of AUCDI and usability assessment or usability capability or usability maturity that involve the following:

"Capability, capability scale/level, capability maturity model, maturity, practice, practice rating, process, process assessment, process capability determination." Basic terms were also defined like "context of use, user experience, user interface, process improvement, software, software developer/software engineer, task, task analysis, usability, usability testing, user, UCD, UCD practitioner, work products". Those constructs are defined in Appendix K, section K.3.

9.5 AUCDI Maturity Model Evolution

This section discusses how the result of the AUCDI maturity model expert evaluation led to its evolution into a number of subsequent versions. The changes to the model's maturity levels, key practices, scoring scheme and the reasons behind these changes were recorded and analysed. The maturity model version, date of change, expert reviewer details, and changes made to the model and their underlying rationale are shown below.

Version	Date	Reviewer
1.0	7/3/2013	Dr. Paul Cairns
2.0	21/4/2014	Mona Singh
2.1	23/4/2014	Jim Ungar
2.2	30/4/2014	Jason Chong Lee

Table 9.5: Evolution of AUCDI Maturity Model

The valuable evaluation that was received from expert reviewers were used to refine the model as follows:

Version 1.0:

Based on feedback from PhD Supervisor, Dr. Paul Cairns, the questions in AUCDI domain expert evaluation forms were transfered into statements, using simpler terms in question wording, and adding questions to elaborate on reviewer's answers.

Version 2.0 to 2.2

Based on feedback from the AUCDI domain experts the following changes were made:

- Maturity level 5 description was updated to include a process for reviewing and assessing guidelines.
- Training practices were updated to include UCD awareness training to product owner. Also further product owner features were added to the people dimension. Those included understanding of UCD and understanding of UCD practitioner role.
- The description of standards, patterns, and style guides was edited to indicate their role in ensuring consistency across products and re-usability as well as their importance in improving projects developed by small agile teams.

- Early work phase description was edited to refer to the communication between UCD practitioners and architects to check the project's technical feasibility. Moreover, the description was edited to include its utilisation in acquiring feedback from management and sales department.

9.6 Conclusion

This chapter reported on the development of a lightweight, descriptive maturity model for integrating Agile development processes and UCD. The maturity model addresses the specifics, requirements, activities, success factors, and challenges identified within the AUCDI domain. This AUCDI maturity model can be used for both process definition and process assessment. Process definition is embodied via providing organisations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI. This AUCDI maturity model provides organisations with a profound and thorough understanding of AUCDI specifics, activities, roles, timing, responsibilities, success factors, and challenges.

Process assessment focuses on providing organisations with a diagnostic tool to assess both the capability and performance for AUCDI. The process assessment results in identifying AUCDI weaknesses and strengths. The results of this assessment can be communicated to: first, management to provide them with a better understanding of the issues involved in consistently developing products with high and competitive usability level as well as pinpoint AUCDI hindrance. Second, developers to provide them with a better understanding of usability and UCD. Third, UCD practitioners by pinpointing areas that require improvement in usability processes and practices.

The AUCDI maturity model is up to our knowledge the first maturity model to approach the process of AUCDI maturation and its constituents. The maturity model is composed of three components: first, a multidimensional reference model that includes a set of fundamental elements that affect AUCDI and thus should be be reflected in the model and examined in an assessment. These dimensions are: UCD infrastructure; AUCDI process; people; and UCD continuous improvement. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. The closer the organisation meets the requirements of the AUCDI reference model, the higher

its ratings. Third, an assessment procedure to provide practical guidance for performing the assessment. The assessment procedure is composed of a performance scale, maturity recording sheet, maturity levels, typical quotes, and assessment guidelines.

9.7 Chapter Summary

This chapter discussed the design and development an AUCDI maturity model. The following chapter will present thesis conclusions and future work.

Chapter 10

Conclusions and Future Work

This thesis investigated the integration of Agile software development processes and UCD in the context of usability maturity models. In doing so it contributed a SLR, an empirical study of utilisation of two UMMs on five AUCDI case studies and a descriptive maturity model for integrating Agile development processes and user centred design.

In this chapter a summary of the research objectives and main contributions is provided. In addition, research limitations are highlighted and suggestions for future work.

10.1 Objectives of Research

The thesis identified and defined a number of research objectives, specifically:

1. To conduct a systematic literature review to investigate Agile and UCD integration challenges, practices and success factors.
2. To conduct an empirical study to investigate industrial AUCDI attempts, which can be used to verify and complement systematic literature review findings.
3. To investigate the suitability of Usability Maturity Models for assessing usability maturity *in the context* of Agile development processes.
4. To Investigate the existence of a relationship between the success of AUCDI attempts and usability maturity levels.
5. To develop a lightweight, descriptive maturity model for integrating Agile development processes and UCD.

10.2 Contributions of the Thesis

This research contributes to the body of research from a variety of perspectives including a SLR and a set of empirical studies as follows. Chapter 3 reported the results of the systematic literature review that explicated AUCDI challenges, practices and success factors. Chapter 4 reported the results of an empirical study of industrial AUCDI attempts that verified and complemented findings of the SLR. Chapters 6 and 7 reported the results of an empirical study that utilised Nielsen Corporate Usability Maturity Model and UMM-HCS maturity model on five AUCDI industrial case studies and demonstrated the value, potential and weaknesses of using existing UMMs in the AUCDI domain. The results, lessons learned and observations from chapters 3, 4, 6, and 7 were utilised in formalizing a set of dimensions for integrating Agile and UCD. These dimensions were used as the basis for formalizing an AUCDI maturity model that was described in chapter 8 and 9 respectively.

10.2.1 AUCDI Systematic Literature Review

The first research contribution was presented in chapter 3, a state of art SLR that identified and classified various challenging factors that restrict AUCDI and explored the proposed practices and success factors to deal with these challenges. The SLR included a total of 71 papers and excluded 80 papers that were published from the year 2000 till 2012. The SLR findings were quantitatively and qualitatively classified and a description and taxonomy of AUCDI challenges and its respective success factors and practices were reported.

This SLR is considered as a contribution to both academic researchers and industrial practitioners. Industrial practitioners can utilise the description and taxonomy of AUCDI challenges and corresponding practices and success factors in identifying potential challenges of AUCDI and practices or success factors to deal with these anticipated challenges. By contrast, academic researchers and industrial practitioners can benefit from this SLR in identifying research areas that exhibit apparent scarcity and thus need to be further exploited via future research work.

10.2.2 Conducting an Empirical Study of Industrial Practices for AUCDI

The second research contribution was presented in chapter 4, an interview study that involved 14 participants from 11 companies in five different countries that investigated industrial AUCDI attempts in order to verify and complement literature findings. These interviews identified the difficulties that hinder industrial AUCDI attempts and the integration practices and success factors; and confirmed and complemented findings of other researchers from the systematic literature review.

10.2.3 Investigating Usability Maturity Models Role in the AUCDI Domain

The third research contribution was presented in chapters 6 and 7. These chapters reported on five one-to-one semi structured interviews that evaluated the suitability of UMMs for utilisation in assessing usability maturity levels in the context of Agile projects and investigated the existence of

a relationship between the success of AUCDI attempts and usability maturity levels. This occurred via utilising two UMMs: Nielsen Model and Usability Maturity Model-Human Centrdness Scale (UMM-HCS) in five AUCDI case studies and assessing their usability maturity levels.

Findings, observations and lessons learned from chapters 6 and 7 revealed that both models are deficient in their theoretical foundations with respect to maturation, scoring scheme, advice on the assessment of criteria, terminology used and accuracy of some of the key practices.

The investigation of the existence of a relationship between the success of AUCDI attempts and usability maturity level via Nielsen model gave an indication of the existence of a correlation between the success of AUCDI attempts and the AUCDI case study's usability maturity level. Whereas the investigation via UMM-HCS revealed that it was not possible to achieve this aim due to the model's linear upgrading which led to discarding considerable achieved attributes by the five case studies. Applying UMM-HCS on the five case studies gave an indication that the linear model of upgrading is contradictory to how organisation's perform since an organisation can score high in some of the practices related to a high maturity level even if the organisation is at a low maturity level.

This study revealed that although both Nielsen model and UMM-HCS do not conflict with Agile values and principles yet both models do not address the specifics, activities, success factors and challenges identified within the AUCDI domain and subsequently they do not pinpoint all dimensions and practices involved in the AUCDI process. Both models do not provide details on the timing, or on how UCD practices can be applied within an Agile development life cycle (or even within sprints), or the lightweight method of applying the different UCD practices along the Agile development life cycle iterations or sprints. These issues need to be taken into consideration by any researcher who considers to develop a usability maturity model for the Agile domain. AUCDI challenges that were not approached by either models are practices regarding the communication, coordination and collaboration between UCD practitioners and Agile developers in order to synchronize and complete their work, practices related to design modularization and chunking, UCD practitioner workload, and maintaining communication between the customer and the development team. Another issue that needs to be approached by the developers of UMMs for the Agile domain is the features and activities of some team roles including XP coach, Scrum master, product owner and whose role can impact the integration process.

10.2.4 Developing a set of Dimensions for AUCDI

The fourth research contribution was presented in chapter 8, the results of the studies described in chapters 3, 4, 6 and 7 contributed to the development of a set of dimensions for integrating Agile development processes and UCD. AUCDI is dependent on four main dimensions: UCD infrastructure that lays the necessary foundation for integration; AUCDI process that includes detailed activities, work products, work flows, roles, and responsibilities; people involved in the integration process; and UCD continuous improvement.

- **UCD Infrastructure**

  UCD infrastructure involves a number of organisational elements that need to be available in order to achieve successful integration of Agile development processes and UCD. UCD infrastructure as a dimension is composed of funds, staff, tools, methods, management support, training, utilisation of standards, patterns and style guides and colocation of developers and UCD practitioners.

- **AUCDI Process**

  Although UCD infrastructure lays the necessary foundation for achieving the integration between Agile development processes and UCD, however, the success of AUCDI is dependent on a number of other dimensions. The second dimension is the presence of an AUCDI process that embraces the iterative and incremental nature of the Agile development process. The AUCDI process is a method independent process that covers AUCDI related activities throughout the Agile development life cycle. The AUCDI process includes detailed activities, work products, work flows, roles, and responsibilities. The proposed AUCDI process is based on the ISO 13407 UCD activities and UCD processes of KESSU 2.2 [147]. However, the AUCDI process extends those foundational UCD processes, activities and principles by addressing the specifics, activities, success factors and challenges identified within the AUCDI domain. The AUCDI process is based around a number of processes that focus on planning and implementation of UCD principles and activities into the Agile development life cycle. Those processes are achieved via the implementation of a set of practices.

  The AUCDI process can be utilised for training purposes to emphasis and highlight the importance of the activities and principles of user-

centred design, make the essence of UCD in the context of Agile projects understandable to development staff and provide both Agile developers and UCD practitioners with a road map on their roles and their interaction. The AUCDI process model clarifies UCD practitioners' roles and responsibilities in development projects and makes them highly visible.

- **People**

  Another dimension that contributes to AUCDI is the people involved in the process. The AUCDI process involves a number of team members including customers, users, developers, UCD practitioners and XP coach in case of utilising XP and Scrum master and product owner in case of utilising Scrum. Successful AUCDI requires communication, collaboration and cooperation between team members. Moreover, successful AUCDI requires particular attributes, for example, skills and awareness in each of those team members.

- **UCD Continuous Improvement**

  The UCD continuous improvement dimension is focused on monitoring the UCD process across projects; systematically improving UCD activities, tools, methods, skills and awareness; benchmarking product's usability and/or user experience against competitive products' usability and/or user experience; and emerging product decisions from end user and customer studies and targeting it to meet users needs and expectations.

10.2.5 Development of an AUCDI Maturity Model

The fifth research contribution was presented in chapter 9, a lightweight, descriptive AUCDI Maturity Model was developed. The maturity model addresses the specifics, activities, success factors and challenges identified within the AUCDI domain. The AUCDI Maturity Model can be used by organisations for two purposes: AUCDI process definition and assessment.

AUCDI Process Definition

Process definition is embodied via providing organisations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI. This AUCDI maturity model provides organisations with a pro-

found and thorough understanding of AUCDI specifics, activities, roles, timing, success factors and challenges.

AUCDI Process Assessment

Process assessment focuses on providing organisations with a diagnostic tool to assess both the capability and performance for AUCDI. Capability assessment falls on the organisation as a whole and it attempts to identify the extent to which UCD is consistently and systematically implemented in the different projects. Performance assessment on the other hand focuses on effective UCD implementation at individual development projects level. The process assessment results in identifying AUCDI weaknesses and strengths.

The AUCDI maturity model is up to our knowledge the first maturity model to approach the process of AUCDI maturation and its constituents. The maturity model is composed of three components: first, a multidimensional reference model that includes a set of fundamental elements that affect AUCDI and thus should be be reflected in the model and examined in an assessment. These dimensions are: UCD infrastructure; AUCDI process; people; and UCD continuous improvement. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. The closer the organisation meets the requirements of the AUCDI reference model, the higher its ratings. Third, an assessment procedure to provide practical guidance for performing the assessment. The assessment procedure is composed of a maturity recording sheet, maturity levels, typical quotes, and assessment guidelines.

The end result is represented in an assessment report. These results can help organisations assess their current status and the factors that impact their AUCDI process and pinpoint weaknesses and strengths in order to identify improvement areas. The results of this assessment can be communicated to different parties including: first, management to provide them a with better understanding of the issues involved in consistently developing products with competitive and high usability level as well as pinpoint AUCDI hindrance. Second, developers to provide them with better understanding of usability and UCD. Third, UCD practitioners by pinpointing areas that require improvement in usability processes and practices.

10.3 Limitations

The first research limitation was related to the five AUCDI case studies that were utilised in Nielsen and UMM-HCS studies since all case studies represented Scrum projects and as a result further investigation needs to be made with XP projects.

Finally, this research depended on AUCDI literature and interview studies, thus it is based on knowledge of academic researchers and industrial practitioners rather than on first hand observations.

10.4 Future Work

In this section a discussion is provided for a number of potential extensions to the research and preliminary work achieved in those areas.

10.4.1 Applying AUCDI Maturity Model Into Case Studies

The AUCDI Maturity Model will be utilised into case studies in order to identify whether the model has succeeded as a diagnostic tool in depicting all aspects involved in AUCDI, assessing the current situation, identifying the strengths and weaknesses related to AUCDI for the chosen project and providing guidance for the improvement actions of the organisation. The changes to the model, maturity levels, key practices, scoring scheme and assessment report and the reasons behind these changes will be recorded and analysed.

10.4.2 AUCDI Maturity Model Deployment and Maintenance

Future work involves deploying the model through making it available via a web site in order to verify the extent of its generalisability and provide wider acceptance and standardization of the model. The web interface will provide a secured access to the maturity questionnaire and then the answers will be studied to provide the organisation aspiring to integrate Agile and UCD with the assessment results. Moreover, the model will be

maintained via keeping a web based repository to track model development and evolution as the domain knowledge and model understanding broadens and deepens.

10.4.3 Developing the AUCDI Maturity Model from Descriptive into Prescriptive

Maturity models have one of three purposes, descriptive, prescriptive or comparative. These different purposes imply different components and properties that can affect the creation, evaluation and utilisation of the models.

The proposed AUCDI maturity model is a descriptive maturity model that is used as a diagnostic tool to assess the current capabilities of the examined entity against specific criteria. Future work involves evolving this model into a prescriptive maturity model that pinpoints desirable maturity levels and provides specific and detailed improvement guidelines.

10.5 Closing Remarks

The work presented in this thesis has explored the potential of usability maturity models in the context of integrating Agile development processes and UCD and demonstrated a systematic method for assessing the organisational and project strengths and weaknesses related to AUCDI.

This chapter is followed by the references list. It is worth noting that each item included in the reference list is followed by a number or a set of numbers. This signifies the page number(s) where this reference was utilised.

Appendix A

Abbreviations and Acronyms

2D: Two Dimensional

ACM: Association for Computing Machinery

AM: Agile Modeling

API: Application Program Interface

ASD: Adaptive Software Development

AUCDI: Agile and User Centred Design Integration

AUCDIMM: Agile and User Centred Design Integration Maturity Model

AUE: Automated Usability Evaluation

BA: Business Administration

BDD: Behaviour Driven Development

BI: Business Intelligence

BPM: Business Process Management Maturity Model

CS: Computer Science

CD: Compact Disc

CDR: Central Design Record

CEO: Chief Executive Officer

CHI: The ACM SIGCHI Conference on Human Factors in Computing Systems

CMM: Capability Maturity Model

CMS: Content Management Systems

CPS: CompuPharaohs

CS: Computer Science

CS1..CSn: Case Study 1.. Case Sudy n

CSS: Cascading Style Sheets

DATech-UEPA: Procedures for Usability Engineering Process Assessment

DP: Design Principles

DS: Development Services

DSDM: Dynamic Systems Development Method

EBSE: Evidence Based Software Engineering

EUSCs: European Usability Support Centres

ERP: Enterprise Resource Planning System

XSBD: eXtreme Scenario Based Design process

FAQs: Frequently Asked Questions

FDD: Feature Driven Development

GPS: Global Positioning System

GUI: Graphical User Interface

HCD: Human Centred Design

HCI: Human Computer Interaction

HF: Human Factors

HFIPRA: Human Factors Integration Process Risk Assessment

HP: Hewlett Packard

HPA: Humanware Process Assessment

HTML: Hyper Text Markup Language

IA: Information Architect

ICSE: The International Conference on Software Engineering

IDE: Integrated Development Environment

IEE: Institution of Electrical Engineers

IEEE: Institute of Electrical and Electronics Engineers

INUSE Project: A European research project for assuring the usability of interactive systems or web sites

IS: Information System

ISIC: International Student Identity Card

ISO: International Organisation for Standardisation

ISO 9241: Ergonomic requirements for office work with visual display terminals

ISO 13407: Human Centred Design Processes for Interactive Systems

ISO 14915: Design of the User Interface of Multimedia

ISP: Internet Service Provider

KESSU: A research project aiming at implementing user centred design in software development organisations

KPAs: Key Process Areas

LDUF: Little Design Upfront

MS TFS: Microsoft Team Foundation Server

MSF: Microsoft Solutions Framework

MVP: Most Valuable Microsoft Professional

(N/A): Not Applicable

OH: Office Hours

OHD: Orascom Hotels and Development

QA: Quality Assurance

QMMG: Crosby's maturity grid

P1..Pn: Project 1..Project n

PAM Team: Project Assessment Management Team

PT1..PTn: Participant 1..Participant n

PI: Process Improvement

PM: Project Management

PM: Project Manager

PO: Product Owner

PhD: Doctor of Philosophy

PSP: Personal Software Process

QE: Quality Engineers

TaMUiator: Task Model-based Usability Evaluator

TDD: Test Driven Development

R and D: Research and Development

RITE: Rapid Iterative Test and Evaluation

RUP: Rational Unified Process

SBD: Scenario Based Design

SE-HCI: Software Engineering - Human Computer Interaction

SDM: Software Development Methodologies

SLR: Systematic Literature Review

SMEs: Small and Medium Enterprises

SMEs: Subject Matter Experts

SPI: Software Process Improvement

TSM: Total Systems Maturity

TV: Television

TSP: Team Software Process

UCD: User Centred Design

UCDM: User Centred Design Maturity

UCM: Usability Capability Maturity

UE: Usability Engineer

UE: Usability Engineering

UED: User Experience Design

UED: User Environment Design

UEMan: User Evaluation Manager

UI: User Interface

UMM: Usability Maturity Models

UMM-P: Usability Maturity Model-Processes

UMM-HCS: Usability Maturity Model-Human Centredness Scale

ULMM: Usability Leadership Maturity Model

UPA: The User Experience Professionals Association

UPA: Usability Process Assessment

UX: User Experience

VDTs: Visual Display Terminals

VP: Vice President

XP: Extreme Programming

XPnUE: Extreme Programming and Usability Engineering

Appendix B

Quality Assessment for Research Papers

No.	Study	1	2	3	4	5	6	7	8	9	10	11	Total
1	[129]	1	1	1	1	1	1	0	0	0	1	1	8
2	[36]	1	1	0.5	1	0.5	0.5	0	0	0	1	1	6.5
3	[3]	1	1	0.5	1	1	1	1	0	0	1	0	7.5
4	[153]	1	1	0.5	0.5	1	0.5	1	0	0	0.5	1	7
5	[62]	1	1	0.5	1	1	1	1	0	NA	1	1	8.5
6	[23]	1	1	1	1	1	1	1	NA	NA	1	1	9
7	[78]	1	1	1	1	1	NA	NA	NA	NA	NA	1	6
8	[156]	1	1	1	1	1	NA	NA	NA	NA	1	1	7
9	[131]	1	1	1	1	1	1	1	0	0	1	1	9
10	[89]	1	1	1	1	1	1	1	0	0	1	1	9
11	[132]	1	1	1	1	1	1	1	0	0	1	0	8
12	[39]	1	1	1	1	1	1	1	0	1	1	1	10
13	[193]	1	1	0.5	1	0.5	1	0	0	0	1	0	6
14	[84]	1	1	1	1	1	1	1	0	0	1	1	9
15	[83]	1	1	1	1	1	1	0.5	0	0	1	0	7.5
16	[82]	1	1	1	1	1	1	0.5	0	0	1	0	7.5
17	[56]	1	1	1	1	1	1	1	0	0	1	0	8
18	[81]	1	1	1	1	1	1	1	0	0	1	0	8
19	[172]	1	1	1	1	0.5	0.5	0.5	0	0	1	1	7.5
20	[239]	1	1	1	1	0.5	NA	NA	NA	NA	0.5	1	6
21	[53]	1	1	1	1	1	1	1	1	NA	1	1	10
22	[267]	1	1	1	0	0	1	0	0	NA	1	1	6
23	[206]	1	1	1	1	0	0.5	0	0	0	0.5	1	6
24	[261]	1	1	1	1	1	1	1	0	0	1	0	8
25	[185]	1	1	1	1	0.5	NA	NA	NA	NA	0.5	1	6
26	[116]	1	1	1	1	1	0.5	0.5	NA	NA	1	1	8
27	[114]	1	1	1	0.5	1	NA	NA	0	0	0	1	5.5
28	[221]	1	1	1	1	0.5	0	0	0	1	0	0	5.5
29	[85]	1	1	1	0.5	1	1	1	0	0	1	0	7.5
30	[86]	1	1	1	1	1	1	1	0	0	1	1	9
31	[177]	1	1	1	1	1	0	0	0	0	1	1	7
32	[176]	1	1	1	1	1	1	0	0	0	1	1	8
33	[33]	1	1	1	1	1	1	1	0	0	1	1	9
34	[254]	1	1	1	1	0	NA	NA	NA	NA	0.5	1	5.5
35	[120]	1	0.5	1	1	0.5	0.5	0	0	0	0.5	1	6
36	[246]	1	1	1	1	1	1	0	0	0	0.5	0.5	7
37	[198]	1	1	1	1	NA	NA	NA	NA	NA	0.5	1	5.5
38	[199]	0.5	1	1	1	0.5	0	0	0	0	0.5	1	5.5
39	[34]	1	1	1	1	1	1	1	0	0	1	1	9
40	[28]	1	0	0.5	0.5	0.5	0	0	0	0	0	1	3.5
41	[115]	1	0.5	1	1	0	0	0	0	0	0	0	3.5
42	[180]	1	1	0	0	0	0	0	0	0	0	0	2
43	[60]	0.5	0.5	0.5	0	0	0	0	0	0	0	0	1.5
44	[91]	0	0	1	0	0	0	0	0	0	0	0	1
45	[284]	1	0.5	0.5	0	0	0.5	0	0	0	0	0	2.5

Table B.1: Quality Assessment for Research Papers

Appendix C

Quality Assessment for Experience Reports

It is important to note that although two papers [59, 228] are evidenced theoretical papers yet they are based on industrial experience and that is the reasons they were included among experience reports.

No.	Study	1	2	3	4	5	Total
1	[262]	1	1	0.5	1	1	4.5
2	[35]	1	1	0.5	1	1	4.5
3	[283]	1	1	1	1	1	5
4	[32]	1	1	1	1	1	5
5	[5]	1	1	1	1	1	5
6	[9]	1	0.5	1	1	1	4.5
7	[41]	1	1	1	1	1	5
8	[200]	1	1	1	1	1	5
9	[76]	1	1	1	1	1	5
10	[278]	1	1	1	1	1	5
11	[113]	1	1	1	1	1	5
12	[209]	1	1	1	1	1	5
13	[59]	1	1	1	1	0	5
14	[275]	1	1	1	1	1	5
15	[183]	1	1	1	1	1	5
16	[181]	1	1	1	1	1	5
17	[226]	1	0.5	1	0.5	0.5	3.5
18	[228]	1	1	1	1	1	5
19	[45]	1	1	1	1	1	5
20	[203]	1	1	1	1	1	5
21	[136]	1	1	1	1	1	5
22	[184]	1	1	1	0.5	0.5	4
23	[272]	1	1	1	1	1	5
24	[178]	1	1	1	1	1	5

Table C.1: Quality Assessment for Experience Reports

Appendix D

Included Papers

1. Adikari, S., C. McDonald, and J. Campbell (2009). Little Design Up-Front: A Design Science Approach to Integrating Usability into Agile Requirements Engineering. In J. Jacko (Ed.), Human-Computer Interaction. New Trends, Volume 5610 of Lecture Notes in Computer Science, pp.549 - 558. Springer Berlin / Heidelberg.
2. Albisetti, M. (2010). Launchpad's Quest for a Better and Agile User Interface. In 11th International Conference on Agile Software Development, XP2010, Trondheim, Norway.
3. Armitage, J. (2004, January). Are Agile Methods Good for Design? Interactions 11(1), pp. 14 - 23.
4. Ambler, S. (2008). Tailoring Usability into Agile Software Development Projects. In E. Law, E. Hvannberg, and G. Cockton (Eds.), Maturing Usability, Human Computer Interaction Series, pp. 75 - 95. Springer. London.
5. Beyer, H., K. Holtzblatt, and L. Baker (2004). An Agile Customer-Centred Method: Rapid Contextual Design. In XP/AU.
6. Blomkvist, S. (2005). Towards a Model for Bridging Agile Development and User-Centred Design. In A. Seffah, J. Gulliksen, and M. Desmarais (Eds.), Human-Centred Software Engineering- Integrating Usability in the Software Development Life cycle, Volume 8 of Human Computer Interaction Series, pp. 219 - 244. Springer. Netherlands.
7. Broschinsky, D. and L. Baker (2008). Using Persona with XP at LANDesk

Software, an Avocent Company. In Proceedings of the Agile 2008, Agile '08, Washington, DC, USA, pp. 543 - 548. IEEE Computer Society.

8. Brown, J., G. Lindgaard, and R. Biddle (2008). Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artefacts. In Proceedings of the Agile 2008, Agile '08, Washington, DC, USA, pp. 39 - 50. IEEE Computer Society.

9. Brown, J., G. Lindgaard, and R. Biddle (2011, Aug.).Collaborative Events and Shared Artefacts: Agile Interaction Designers and Developers Working Toward Common Aims. In Agile Conference, 2011, pp. 87 - 96.

10. Budwig, M., S. Jeong, and K. Kelkar (2009). When User Experience Met Agile: A Case Study. In Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems, CHI EA '09, New York, NY, USA, pp. 3075 - 3084. ACM.

11. Carbon, R., J. Dorr, and M. Trapp (2004). Focusing Extreme Programming on Usability. In GI Jahrestagung, pp. 147 - 152.

12. Chamberlain, S., H. Sharp, and N. Maiden (2006). Towards a Framework for Integrating Agile Development and User Centred Design. In Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering, XP'06, Berlin, Heidelberg, pp. 143 - 153. Springer-Verlag.

13. Coatta, T. and R. Rutter (2011). UX Design and Agile: A Natural Fit? Communications of the ACM 54(1), pp. 54 - 60.

14. Constantine, L. and L. Lockwood(2001b). Process Agility and Software Usability: Towards Lightweight Usage Centred Design. Information Age 8.8

15. Dayton, D. and C. Barnum (2009). The Impact of Agile on User-Centred Design: Two Surveys Tell the Story. Technical Communication 56(3), pp. 219 - 234.

16. Federoff, M. and C. Courage (2009). Successful User Experience in an Agile Enterprise Environment. In Proceedings of the Symposium on Human Interface 2009 on Conference Universal Access in Human-Computer Interaction. Part I: Held as Part of HCI International 2009, Berlin, Heidelberg, pp. 233 - 242. Springer-Verlag.

17. Da Silva, T. S., A. Martin, F. Maurer, and M. Silveira (2011, Aug.). User-Centred Design and Agile Methods: A Systematic Review. In Agile Conference (Agile), 2011, pp. 77 - 86.

18. Detweiler, M. (2007, May). Managing UCD within Agile Projects. Interactions 14(3), pp. 40 - 42.

19. Duchting, M., D. Zimmermann, and K. Nebe (2007). Incorporating User Centred Requirement Engineering into Agile Software Development. In Proceedings of the 12th international Conference on Human-Computer Interaction: Interaction Design and Usability, HCI'07, Berlin, Heidelberg, pp. 58 - 67. Springer-Verlag.

20. Ferreira, J. (2008). Interaction Design and Agile Development: Reconciling Iterative and Incremental Approaches. In Agile Conference.

21. Ferreira, J., J. Noble, and R. Biddle (2007a). Agile Development Iterations and UI Design. In Proceedings of the Agile 2007, Agile'07, Washington, DC, USA, pp. 50 - 58. IEEE Computer Society.

22. Ferreira, J., J. Noble, and R. Biddle (2007b). Interaction Designers on eXtreme Programming Teams: Two Case Studies from the Real World. In Proceedings of the Fifth New Zealand Computer Science Research Student Conference (NZCSRSC2007).

23. Ferreira, J., J. Noble, and R. Biddle (2007c). Up-front Interaction Design in Agile Development. In Proceedings of the 8th International Conference on Agile Processes in Software Engineering and Extreme Programming, XP'07, Berlin, Heidelberg, pp. 9 - 16. Springer-Verlag.

24. Ferreira, J., H. Sharp, and H. Robinson (2010). Values and Assumptions Shaping Agile Development and User Experience Design in Practice. In A. Sillitti, A. Martin, X. Wang, E. Whitworth, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, and C. Szyperski (Eds.), Agile Processes in Software Engineering and Extreme Programming, Volume 48 of Lecture Notes in Business Information Processing, pp. 178 - 183. Springer Berlin Heidelberg.

25. Ferreira, J., H. Sharp, and H. Robinson (2011, August). User Experience Design and Agile Development: Managing Cooperation through Articulation Work. Software Practice and Experience 41(9), pp. 963 - 974.

26. Ferreira, J., H. Sharp, and H. Robinson (2012). Agile Development and User Experience Design Integration as an On-going Achievement in Practice. In Agile 2012.

27. Fox, D., J. Sillito, and F. Maurer (2008). Agile Methods and User-Centred Design: How These Two Methodologies are Being Successfully Integrated in Industry. In Proceedings of the Agile 2008, Agile '08, Washington, DC, USA, pp. 63 - 72. IEEE Computer Society.

28. Hodgetts, P. (2005). Experiences Integrating Sophisticated User Experience Design Practices into Agile Processes. In Proceedings of the Agile Development Conference, ADC '05, Washington, DC, USA, pp. 235 - 242. IEEE Computer Society.

29. Patton, J. (2002b). Hitting the Target: Adding Interaction Design to Agile Software Development. In OOPSLA 2002 Practitioners Reports, OOPSLA 2002, New York, NY, USA. ACM.

30. Holzinger, A., M. Errath, G. Searle, B. Thurnher, and W. Slany (2005). From Extreme Programming and Usability Engineering to Extreme Usability in Software Engineering Education. In Proceedings of the 29th Annual International Conference on Computer Software and Applications Conference, COMPSAC-W'05, Washington, DC, USA, pp. 169 - 172. IEEE Computer Society.

31. Hosseini-Khayat, A., T. Hellmann, and F. Maurer (2010, aug.). Distributed and Automated Usability Testing of Low-Fidelity Prototypes. In Agile Conference (Agile), pp. 59 - 66.

32. Hudson, W. (2003). Adopting User-Centred Design within an Agile Process: A Conversation. Cutter IT Journal 16.10, pp. 5 - 12.

33. Humayoun, S. R., Y. Dubinsky, and T. Catarci (2011). A Three-Fold Integration Framework to Incorporate User-Centred Design into Agile Software Development. In Proceedings of the 2nd International Conference on Human Centred Design, HCD'11, Berlin, Heidelberg, pp. 55 - 64. Springer-Verlag.

34. Hussain, Z., W. Slany, and A. Holzinger (2009b). Investigating Agile User-Centred Design in Practice: A Grounded Theory Perspective. In Proceedings of the 5th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for e-Inclusion, USAB '09, Berlin, Heidelberg, pp. 279 - 289. Springer-Verlag.

35. Hussain, Z., W. Slany, and A. Holzinger (2012). Agile Software Development Methods and Usability/ User Centred Design: Perspectives from an Online Survey. Journal of Systems and Software.

36. Hussain, Z., H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, T. Vlk, C. Koeffel, M. Tscheligi, and P. Wolkerstorfer (2012). Practical Usability in XP Software Development Processes. In The Fifth International Conference on Advances in computer Human Interactions, ACHI 2012.

37. Illmensee, T. and A. Muff (2009). 5 Users Every Friday: A Case Study in Applied Research. In Proceedings of the 2009 Agile Conference, Agile '09, Washington, DC, USA, pp. 404 - 409. IEEE Computer Society.

38. Jokela, T. and P. Abrahamsson (2004). Usability Assessment of an Extreme Programming Project: Close Co-operation with the Customer Does Not Equal to Good Usability. In F. Bomarius and H. Iida (Eds.), Product Focused Software Process Improvement, Volume 3009 of Lecture Notes in Computer Science, pp. 393 - 407. Springer Berlin / Heidelberg.

39. Kane, D. (2003). Finding a Place for Discount Usability Engineering in Agile Development: Throwing Down the Gauntlet. In Proceedings of the Conference on Agile Development, ADC 2003, Washington, DC, USA. IEEE Computer Society.

40. Kollmann, J. (2008). Designing the User Experience in an Agile Context. Masters thesis, University College London.

41. Lee, J. C. and S. McCrickard (2007). Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. In Proceedings of the Agile 2007, Agile '07, Washington, DC, USA, pp. 59 - 71. IEEE Computer Society.

42. Lee, J. C., S. McCrickard, and T. Stevens (2009, Aug.). Examining the Foundations of Agile Usability with eXtreme Scenario-Based Design. In Agile Conference, pp. 3 - 10.

43. Lee, J. C., T. Judge, and S. McCrickard (2011). Evaluating EXtreme Scenario-Based Design in a Distributed Agile Team. In Proceedings of the 2011 Annual Conference on Human factors in Computing Systems, CHI EA'11, New York, NY, USA, pp. 863 - 877. ACM.

44. Leszek, A. and C. Courage (2008, aug.). The Doctor is "In" Ű Using

the Office Hours Concept to Make Limited Resources Most Effective. In Agile Conference 2008, pp. 196 - 201.

45. Lievesley, M. and J. Yee (2006). The Role of the Interaction Designer in an Agile Software Development Process. In CHI'06 Extended Abstracts on Human Factors in Computing Systems, CHI EA Š06, New York, NY, USA, pp. 1025 - 1030. ACM.

46. Lindstrom, H. and M. Malmsten (2008). User-Centred Design and Agile Development: Rebuilding the Swedish National Union Catalogue. The Code4Lib Journal 5, pp. 12 - 15.

47. Losada, B., M. Urretavizcaya, and I. Fernandez-deCastro (2011). Agile Development of Interactive Software by means of User Objectives. In The Sixth International Conference on Software Engineering Advances.

48. Larusdottir, M., E. Bjarnadottir, and J. Gulliksen (2010). The Focus on Usability in Testing Practices in Industry. In P. Forbrig, F. Patern, and A. Mark Pejtersen (Eds.), Human-Computer Interaction, Volume 332 of IFIP Advances in Information and Communication Technology, pp. 98 - 109. Springer Boston.

49. McInerney, P. and F. Maurer (2005, November). UCD in Agile Projects: Dream Team or Odd Couple? Interactions 12(6), pp. 19 - 23.

50. Mcneil, M. (2006). Agile User-Centred Design. In Proceedings of the International Conference on Contemporary Ergonomics (CE2006), Cambridge, UK.

51. Meszaros, G. and J. Aston (2006). Adding Usability Testing to an Agile Project. In Proceedings of the conference on Agile 2006, Agile '06, Washington, DC, USA, pp. 289 - 294. IEEE Computer Society.

52. Memmel, T., F.Gundelsweiler, andH.Reiterer(2007). Agile Human-Centred Software Engineering. In Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...but not as we know it - Volume 1, BCS-HCI '07, Swinton, UK, UK, pp. 167 - 175. British Computer Society.

53. Memmel, T., H. Reiterer, and A. Holzinger (2007). Agile Methods and Visual Specification in Software Development: A Chance to Ensure Universal Access. In Proceedings of the 4th International Conference on Universal Access in Human Computer Interaction: Coping with Diversity, UAHCI'07, Berlin, Heidelberg, pp. 453 - 462.

Springer-Verlag.

54. Miller, L. (2005). Case Study of Customer Input For a Successful Product. In Proceedings of the Agile Development Conference, ADC '05, Washington, DC, USA, pp. 225 - 234. IEEE Computer Society.

55. Moreno, A. and A. Yagie (2012). Agile User Stories Enriched with Usability. In C. Wohlin (Ed.), Agile Processes in Software Engineering and Extreme Programming, Volume 111 of Lecture Notes in Business Information Processing, pp. 168 - 176. Springer Berlin Heidelberg.

56. Najafi, M. and L. Toyoshiba (2008). Two Case Studies of User Experience Design and Agile Development. In Proceedings of the Agile 2008, Agile '08, Washington, DC, USA, pp. 531 - 536. IEEE Computer Society.

57. Obendorf, H. and M. Finck (2008). Scenario-Based Usability Engineering Techniques in Agile Development Processes. In CHI '08 Extended Abstracts on Human Factors in Computing Systems, CHI EA '08, New York, NY, USA, pp. 2159 - 2166. ACM.

58. Parsons, D., R. Lal, and H. Ryu (2007). Software Development Methodologies, Agile Development and Usability Engineering. In 18th Australian Conference on Information Systems.

59. Peixoto, C. S. A. and A. E. A. da Silva (2009). A Conceptual Knowledge Base Representation for Agile Design of Human-Computer Interface. In Proceedings of the 3rd international Conference on Intelligent Information Technology Application, IITA'09, Piscataway, NJ, USA, pp. 156 - 160. IEEE Press.

60. Rannikko, P. (2011, April). User Centred Design in Agile Software Development. Master's thesis, University of Tampere, School of Information Science.

61. Rittenbruch, M., G. McEwan, nigel Ward, T. Mansfield, and D. Bartenstein (2002). Extreme Participation - Moving Extreme Programming Towards Participatory Design. In Proceedings of the Seventh Biennial Participatory Design Conference.

62. Salah, D., H. Petrie, and R. Paige (2009). Towards a Framework for Bridging User-Centred Design and Agile Software Development Processes. In 3rd Irish HCI Conference 2009.

63. Salah, D. (2011, May). A Framework for the Integration of User Centred Design and Agile Software Development Processes. In 33rd

International Conference on Software Engineering (ICSE), pp. 1132 - 1133.

64. Sharp, H., H. Robinson, and J. Segal (2004). Integrating User-Centred Design and Software Engineering: a Role for eXtreme Programming? In BCS-HCI Group's 7th Educators Workshop: Effective Teaching and Training in HCI.

65. Singh, M. (2008, aug.). U-SCRUM: An Agile Methodology for Promoting Usability. In Agile Conference, pp. 555 - 560.

66. Sohaib, O. and K. Khan (2010, june). Integrating Usability Engineering and Agile Software Development: A Literature Review. In Computer Design and Applications (ICCDA), 2010 International Conference on, Volume 2, pp. V2 - 32 - V2 - 38.

67. Sy, D. (May 2007). Adapting Usability Investigations for Agile User-Centred Design. Journal of Usability Studies 2(3), pp. 112 - 132.

68. Sy, D. and L. Miller (2008). Optimizing Agile User Centred Design. In CHI '08 Extended Abstracts on Human Factors in Computing Systems, CHI EA '08, New York, NY, USA, pp. 3897 - 3900. ACM.

69. Ungar, J. and J. White (2008). Agile User Centred Design: Enter the Design Studio - A Case Study. In CHI '08 Extended Abstracts on Human Factors in Computing Systems, CHI EA '08, New York, NY, USA, pp. 2167 - 2178. ACM.

70. Tzanidou, K. and J. Ferreira (2010). Design and Development in the Agile Room. Trialing Scrum at a Digital Agency. In A. Sillitti, A. Martin, X. Wang, and E. Whitworth (Eds.), Agile Processes in Software Engineering and Extreme Programming, Volume 48 of Lecture Notes in Business Information Processing, pp. 372 - 378. Springer Berlin Heidelberg.

71. Williams, H. and A. Ferguson (2007). The UCD Perspective: Before and After Agile. In Proceedings of the AGILE 2007, AGILE '07, Washington, DC, USA, pp. 285 - 290. IEEE Computer Society.

Appendix E

Interview Questions

Category 1: Company and Interviewee

1. Where is your company located?
2. What is your job role? Do you perform multiple job roles?
3. How many years have you been working in Agile software development methods? Which Agile methods did you work with? Do you have specific experience in UCD?

Category 2: Project

4. Could you give a brief description on an Agile project that you worked on, which placed high importance on UCD, emphasising its duration and team structure?
5. Which Agile method did you use in this project?

Category 3: Requirements

6. What are the techniques used by the project's team for requirement elicitation?
7. What were the usability goals for this project?
8. What were the user experience goals for this project?
9. Did you gather user interface requirements? How? When? What techniques did you use? How long did this phase take?

Category 4: Agile Process

10. What were the Agile practices adopted in your project? Did you modify or drop any of these practices to achieve your usability or user experience goals? How? Did you add any practices to achieve your usability or user experience goals? How?

11. What tools did you use to support your Agile process in this project? How do you evaluate these tools in regards to enabling you to achieve your requirements in general and your UCD requirements in specific? (Did it enable you to achieve your usability/ user experience goals)? How?

Category 5: Agile and UCD Integration Issues

12. Who is responsible for ensuring the achievement of usability goals in this project's development team? Who is responsible for ensuring the achievement of user experience goals in this project's development team? Where are they located?

13. Can you describe an iteration of your Agile process? When, during the iteration did user interface implementation occur? How long is your iteration? How long does work on user interface implementation take?

14. What were the communication methods and tools used for conveying information between the development team and UCD practitioners or the team member playing their role? How do you evaluate those methods and tools?

Category 6: Testing

15. What were the techniques used for usability testing? When did it occur (how does it fit into the iteration)? How long did it take? How many participants were involved in it? How did you select those participants?

16. When did you implement user feedback for the results of the user interface evaluations? How long did it take? When did you retest the software? How long did it take? How many participants were involved in it? How did you select those participants?

Category 7: General Questions to Wrap Up

17. Do you consider your model as an Agile UCD model? What are the advantages and disadvantages of adopting Agile user-centred design on your Agile process?
18. What worked well and what could be improved in adopting AUCDI in your Agile process?
19. Are there any comments that you would like to add?

Appendix F

Participants Profiles

Participant 1

The first participant worked as a technical team leader whose responsibility was: leading the technical team in validating requirements, design and development, communicating risks and problems to project manager, and communicating with quality team. He also acted as a senior software developer. He was awarded Most Valuable Microsoft Professional (MVP) in 2008.

He has 7 years of total experience in software development and one year experience in Agile software development projects. He graduated from faculty of engineering, computer science department. He has no experience in UCD since he felt it was "not his responsibility", since the development team has a graphics designer and a usability engineer.

Participant 2

The second participant worked as a team leader and a system analyst. She acted as the contact person between developers and customers. She is responsible of acquiring requirements and formatting it in an understandable way to developers and following up with customers to clarify any issues related to the requirements that are raised by developers.

She has three and half years of total experience in software development and one and half years of experience in Agile software development projects. She graduated from Faculty of computers, Ain Shams University, computer science department. She has no experience in UCD from her bachelor degree yet she gained a lot of experience from practice and from customers.

Participant 3

The third participant worked as a technical architect and acted in the project as a technical leader, project manager and Agile coach.

He was awarded Most Valuable Microsoft Professional in 2008.

He has 10 years of total experience in software development and two years experience in Agile software development projects. He graduated from Sadat Academy for Management Science, Information Systems and Computer Science Department. He has no experience in UCD, because his "focus as an architect was more on the back end and the software design itself not on the usability of the system".

Participant 4

The fourth participant worked as a principal technical consultant.

He was awarded Most valuable Microsoft Professional in Microsoft team system for three consecutive years.

He has 14 years of total experience in software development and 7 year experience in Agile software development projects. He graduated from Sadat Academy for Management Science, Information Systems and Computer Science Department. He had a PhD in information systems. He has no experience in his bachelor in UCD yet he studied UCD through interactive CDs.

Participant 5

The fifth participant worked as an independent technical consultant. He acted in this project as the team leader, architect and developer.

He was awarded Most Valuable Microsoft Professional in 2008 in Microsoft Silverlight, a platform to empower building rich user experience web applications. Silverlight allows for integrating video experience, rich graphics, animation, sound, documents.

He has 10 years of total experience in software development and two years experience in Agile software development projects. He graduated from faculty of engineering, computer science department. He has no experience in UCD but gained some through his experience with Silver Light.

Participant 6

The sixth participant worked as a lead software engineer. His job role was technical follow up on design and coding. He contributes to design, code

reviews, setting coding guidelines and acts as an Agile coach interchangeably with the project lead. He was awarded Most Valuable Microsoft Professional (MVP) in 2008.

He has 8 years of total experience in software development and two years experience in Agile software development projects. He graduated from faculty of engineering as a structural engineer but he made a diploma in Computer Science (CS) and is doing a masters in computer science at the Arab Academy for Science, Technology and Maritime Transport.

He contributes to UCD, however, since his team develops off the shelf products, thus his work does not involve any communication with end users. Another team is responsible for product customization and communication with users. He gained experience in UCD through work practices.

Participant 7

The seventh participant worked as a senior developer.

He has five years of total experience in software development and four years of experience in Agile software development projects. He graduated from 6th of October University, Computer Science Department. He does not have any specific experience with UCD.

Participant 8

The eighth participant worked as the interaction design group leader within the user experience team. He has two years of total experience in software development and one and half year experience in Agile software development projects. He was trained as a designer, product developer or industrial designer then he worked as a web designer. He did two years of postgraduate studies in user system interaction.

Participant 9

The ninth participant worked as a level one software engineer. He has one and half years of total experience in software development and one and half years of experience in Agile software development projects. He has no experience in UCD whether from bachelor degree or work.

Participant 10

The tenth participant worked as the product manager and acted also as the customer proxy. He has 17 years of total experience in software development and 13 years of experience in Agile software development projects,

since the first Agile project he did was 13 years ago, even before the term Agile was coined. He graduated from department of computer science and has a Masters degree in business administration.

He has no experience in UCD since this has never been his job role - (although he is acting as an Agile coach and an Agile instructor) - and he learned UCD from reading books, for example, Allan Cooper book "The Inmates are Running the Asylum". " He has a bachelor degree in computer science.

Participant 11

The eleventh participant worked as a product manager. She is mainly responsible for education projects.

She has 12 years of total experience in software development and 6 years of experience in Agile software development projects. She graduated from Sadat Academy for Management Science, Information Systems and Computer Science Department. She has a Masters degree in business administration. She has no experience in UCD.

Participant 12

The twelfth participant worked as a vice president of technology and also acted as scrum master.

He has five years of total experience in software development and one year of experience in Agile software development projects. He graduated from Faculty of computer science, Helwan University. He acquired experience in UCD through practice and an HCI course in his pre-masters degree in computer science.

Participant 13

The thirteenth participant worked as a business analyst. She has 15 years of total experience in software development and 8 year of experience in Agile software development projects. She has a bachelor degree and PhD in computing and maths. she gained experience in UCD via university courses, attending tutorial sessions, and reading books.

Participant 14

The fourteenth participant worked as an independent technical consultant who was also acting as the project and product manager.

He has 12 years of total experience in software development and 6 year of experience in Agile software development projects. He has a bachelor

degree in computer science and management. He gained experience in UCD via university courses, reading books and experience on the job.

Appendix G

Projects Profiles

Some of the projects involved developing new software and new UI designs, others were focused on developing new versions of software and existing UI designs. Some projects are finished and others are still ongoing. The following section presents the project profiles studied so far. For the sake of protecting the anonymity of participants, company names and some details on the projects will be withheld.

Project 1

The first project was performed by (C1), a multinational software development company that has its main office in Egypt but has sales offices in Saudi Arabia, Qatar, Dubai and the United States. The project developed a school educational portal for students, teachers and parents. This portal represents a virtual learning environment.

The development team was composed of a project leader, two junior software developers, one team leader, one technical leader, one technical architect, product manager (customer), three graphics designers, one part time usability engineer and four quality engineers. The team leader, technical leader and technical architect all acted as senior software developers as well.

The project duration was four months. The Agile method adopted was scrum.

Project 2

The second project was performed by (C2), a software development company located in Egypt that provides outsourcing activities. The customer is located in Holland and works in yield management.

The project developed a business intelligence application; a web based platform that provides services to marketers and advertisers. This software checks advertising and marketing campaigns for success, revenue and abiding to budget. The web site is updated daily and users can view and analyse statistics and view reports.

The team was composed of one tester, three junior software developers, one database administrator who also acted as a business intelligence specialist, scrum master, team leader who also acted as a system analyst, and a technical leader who was added after 16 months and who also acted as a developer, a non collocated designer who was located in another country.

The project duration is four years; 1.5 years have passed so far. The Agile method adopted was Scrum.

Project 3

The third project was performed by (C3), a software company located in Egypt. The project developed an Enterprise Resource Planning (ERP) system for a large enterprise. The project aimed to develop a custom ERP System for inventory management of construction equipment and building materials. In addition, the software focused on other aspects like project management, customer relationship management and financial management.

The team was composed of two senior software developers, two juniors, one senior tester, and later became two senior testers, one architect who acted as a project manager and a team leader, part time graphics designer, ERP implementer who also acted as a business analyst.

The project duration was supposedly 8 months of development and four months of testing, this was the initial estimate, the project is currently stopped and it has covered the project cost in one iteration (three months). The Agile method adopted was Scrum.

Project 4

The fourth project was performed by (C4), a Microsoft golden partner located in Egypt.

The project developed a business intelligence and data warehousing application. It developed a governmental data warehouse portal and dash board for tracking the Egyptian exports to 192 countries all over the world.

The team was composed of 7 members, principal technical consultant who also acted as an analyst, architect, project manager, designer who was supposed to act as a usability engineer as well yet it ended up that the usability engineer role was played by the principal technical consultant, senior team leader, three BI specialists, web based developer, part time web designer.

The project duration was 10 months. The Agile method adopted was Microsoft Solutions Framework Agile (MSF).

Project 5

The fifth project was performed by an independent consultant who outsourced the project from a Canadian company called (C5).

The project aimed to develop a Project Management (PM) tool for Agile projects, a visual dash board for monitoring and tracking Agile teams' project status. This dash board presents the results of continuous monitoring of Agile teams' progress. It presents the build status, recent check-ins, bug history, team velocity, effort, and percentage of work done versus not done.

The difference between this tool and other Agile project management tools is that it focuses on empowering Microsoft tools developers to work with Agile methodologies. It will be integrated with Microsoft Team Foundation Server (MS TFS) in order to fill the market gap caused by the limited integration of other Agile project management tools; like Version One, with MS TFS. Although TFS Agile contains an Agile process template yet it concentrates more on implementation rather than the Agile mind set. Thus this project tries to fill in the MS TFS gap in order to encourage better adaptation of Agile methodology.

The team was composed of one interaction designer, one software developer, two testers and a team leader who also acted as a developer and architect.

The project duration is 6 months and it started three months ago. The Agile method adopted was Scrumban, a mixture of Scrum and Kanban

Lean Method.

Project 6

The sixth project was performed by (C1), a multinational software development company that has its main office in Egypt but has sales offices in Saudi Arabia, Qatar, Dubai and the United States. The project developed an authoring tool that allows teachers to prepare lessons and questions for students and allows students to interactively access the lessons, submit homework or solve exams.

The team was composed of three software engineers; two part time senior software engineers and a junior software engineer, two quality engineers (QE) and a senior quality engineer who also acted as a QE team leader, lead software engineer, product manager, project manager, part time usability engineer who was responsible for producing wire frames, and a graphics designer who was responsible for animations and colours, and technical writers team.

The project duration was four to five months during which two software releases were produced.

The Agile method adopted was a mixture of Scrum and XP.

Project 7

The seventh project was performed by (C6), a multinational company with a main office in Egypt and development branch in Azrbigan and sales branch in the United States. C6 is specialized in developing Content Management Systems (CMS).

The project developed version two of a software whose first version failed to achieve user satisfaction. The scope of this software was to provide a work flow platform. The software allowed students to apply for training in the United States, get immigration and visa processed via a third party, and book for flight and accommodation. The software integrates with other software systems, for example, ministry of immigration. Then the software monitors the trainee's dependents status, training evaluation feedback and aggregates the results related to either the third party or the trainee.

The team was composed of two senior developers, two junior developers, a tester, a part time UI designer / user experience designer, one part time team leader who was hired later in the project, one business owner who also acted as an analyst and programmer. The scrum master role was iter-

ated on project team members on every sprint and his main responsibility was to solve previous sprint bugs.

The project duration was supposed to finish in 6 months, yet it took 9 months to launch and acquire user approval. Minor updates continued until the end of the year. The Agile method adopted was a mixture of Scrum and XP.

Project 8

The eighth project was performed by (C7), a multinational company for portable Global Positioning System (GPS) car navigation systems, whose head quarters is located in Amsterdam in the Netherlands. C7 also has offices in Europe, Asia and the United States. Research and development is based in Amsterdam. The project developed a new release for the car navigation software.

The team was composed of a product/business owner, UX representative who was also responsible for interaction design, two technical leads, 20-30 developers, 10 testers and company board who represent stakeholders and is not directly involved in development.

The project duration was four to five months and new releases are produced every 9 months. The Agile method adopted was Scrum.

Project 9

The ninth project was performed by (C8), a multinational company located in Canada but owned by an American company and any software is used in the United States. The project developed a system for enforcing permission for purchasing Television (TV) programs interactively via a TV guide.

The team was composed of four people and the project is composed of 200 people. The team was composed of three developers, one quality assurance specialist and a part time designer. This team did not have an interaction designers or usability engineers as part of the team, the interface designer was a committee composed of three product managers located in the United States.

The project duration was one and half years. The Agile method adopted was a mixture of Scrum and XP.

Project 10

The tenth project was performed by (C9). The project developed a software for an internet service provider. The team was composed of a business analyst who also acted as a usability engineer, 7 or 8 experienced developers, one tester, a scrum master project manager, key manager, and three or four network engineers. The project duration was ongoing. The Agile method adopted was a mixture of Scrum and XP.

Project 11

The eleventh project was performed by (C10). The project developed an extranet portal. It was used to provide a self service venue for customers in order to reduce support calls. Customers used the portal to download new versions of software, register support calls, include feedback on support calls, provide Frequently Asked Questions (FAQs), provide documentation, and provide discussion forum. The team was composed of two developers who also acted as testers and a product manager who acted as a proxy customer. The project duration was two and half years. The Agile method adopted was Scrum.

Project 12

The twelfth project was performed by (C11). The project developed an enterprise resource planning system. The project was composed of three separate teams. The team that was the focus of this study was composed of a number of members who played multiple roles: a product owner who also acted as a scrum master, a business analyst who also acted as a project owner, a development lead who also acted as a project manager, a designer who also acted as a usability engineer, a number of developers who acted as testers. The project duration was one year. The Agile method adopted was Microsoft Solutions Framework Agile.

Appendix H

Informed Consent Form for Nielsen Model Empirical Study - Recorded Consent

My name is Dina Salah, I am a PhD student, at the university of York, working in the area of Agile user centred design integration. I am supervised by Professor Richard Paige.

Would you mind if I recorded this interview? This will make it easier for me to concentrate on what you are saying rather than get distracted by taking notes.

The purpose of this research is to study insightful attempts in integrating Agile and user centred design practice.

The data collection will occur through a 1 hour recorded interview via Skype in which I will ask you a number of questions regarding your published attempt in integrating Agile and user centred design practice.

I would like to clarify that your participation in this interview is entirely voluntary and you are free to withdraw at any point and you do not need to give a reason for withdrawal. I would like also to assure you that all data gathered from the interview will be treated in a confidential manner: It will be archived in a secure location and will only be accessed by me and my supervisory team for the purposes of the analysis. When your data are reported or described, all identifying information will be removed. You may also request for destroying the data recording and any transcripts if you wish so at a later date.

There are a number of benefits that can be achieved from participating in this study. Examples of these benefits are getting an understanding of the organisational performance in user centred issues and identifying attitude, technology and management practices that needs to be improved in order to make your organisation more user centred. There are no known risks to participation in this study.

Please feel free to interrupt, or ask for clarification on any of the interview questions. Now if you are willing to participate, please declare so and I will proceed with the interview. Are you willing to participate in the study I just described?

Appendix I

Description of Levels for Usability Maturity Model-Human Centrdness Scale

This appendix includes the description of the practices that typify each maturity level in UMM-HCS model [68]. It is included in this thesis to allow the reader to easily understand how UMM-HCS model was utilised in AUCDI case studies. Further details on UMM-HCS model can be found [68].

Level X: Unrecognised

The need for a human-centred process is not recognised. If systems are received with varying degrees of satisfaction by their end users this does not cause concern. There are no positive human-centred attributes at this level.

Level A: Recognised

The organisation recognises that there is a need to improve the quality in use of its systems. The organisation has a development process and produces systems. Members of the organisation understand the business benefit of producing usable products.

A.1 Problem Recognition Attribute

The extent to which members of the organisation understand that there is a problem with the quality in use of the systems produced. In order to

ID	Practice
A1.1	Problem recognition. Management and staff are aware that there is a need to improve aspects of the systems under development concerned with their use.

Table I.1: Problem Recognition Attribute [68]

ID	Practice
A2.1	Information collection. Information is collected which could be used to take account of user requirements.
A2.2	Performance of relevant practices. Practices are performed which could be used to include information about user requirements in the system or service.

Table I.2: Performed Processes Attribute [68]

achieve this level of maturity an organisation should carry out the following:

A.2 Performed Processes Attribute

The extent to which processes are performed that provide input that could be used to make the system human-centred. In order to achieve this level of maturity an organisation should carry out the following:

Level B: Considered

The organisation makes its staff aware that quality in use is an important attribute of its products and it engages in awareness raising and training to make its staff aware that quality in use can be improved by taking account of end-user requirements during development of the product. The following attributes of the process demonstrate the achievement of this level, in addition to the attributes for the previous level:

B.1 Quality in Use Awareness Attribute

The extent to which the staff carrying out a process are aware of quality in use as an attribute of the system. In order to achieve this level of maturity an organisation should carry out the following:

B.2 User Focus Attribute

The extent to which staff performing processes relating to the user-facing elements of the system take account of the fact that a human being will need to use it. In order to achieve this level of maturity an organisation should carry out the following:

ID	Practice
B1.1	Quality in use training. Staff are made aware that quality in use is a particular attribute of a system which can be improved.
B1.2	Human-centred methods training. Staff are made aware that quality in use is achieved through the use of a series of human-centred processes during the development and support/use of a system.
B1.3	Human-system interaction training. Staff are made aware that human-centredness covers the total system, not just the user interface or the physical ergonomics.

Table I.3: Quality in Use Awareness Attribute [68]

ID	Practice
B2.1	User consideration training. Staff are made aware that the needs of the end users of the system should be considered when developing or supporting the system.
B2.2	Context of use training. Staff are made aware that end users' skills, background and motivation may differ from developers or system support staff.

Table I.4: User Focus Attribute [68]

Level C: Implemented

Human-centred processes are fully implemented and produce good results. End-users or suitable representatives are involved in specifying and testing systems. Suitably trained staff are available as required to perform the processes which take account of user issues. Techniques are employed which are appropriate to the system, stage in the development and the end users. The following attributes of the process demonstrate the achievement of this level, in addition to the attributes for all the previous levels:

C.1 User Involvement Attribute

The extent to which information is elicited from representative users using appropriate techniques throughout the lifecycle. In order to achieve this level of maturity an organisation should carry out the following:

ID	Practice
C1.1	Active involvement of users. The development process ensures understanding of user needs through user involvement in all development phases.
C1.2	Elicitation of user experience. The design solution is shown to stakeholders and they are allowed to perform tasks (or simulated tasks).
C1.3	End users define quality-in-use. Systems are tested using measures of quality in use derived from end users.
C1.4	Continuous evaluation. Early and continual testing is an essential element of the development methodology. The process is based on the necessity for feedback from users.

Table I.5: User Involvement [68]

ID	Practice
C2.1	Provide appropriate human-centred methods. Select and support methods for the elicitation of user input at all stages in the lifecycle.
C2.2	Provide suitable facilities and tools. Suitable facilities and tools are provided for quality in use activities.
C2.3	Maintain quality in use techniques. Ensure that methods and techniques are reviewed for suitability and that state-of-the-art user interface technologies are used as appropriate in developing new systems.

Table I.6: HF Technology Attribute [68]

C.2 HF Technology Attribute

The extent to which human factors methods and techniques are used in or by human-centred processes. In order to achieve this level of maturity an organisation should carry out the following:

C.3 HF Skills Attribute

The extent to which human factors skills are used in human-centred processes. In order to achieve this level of maturity an organisation should carry out the following:

Level D: Integrated

ID	Practice
C3.1	Decide on required skills. Identify required competencies and plan how to make these available in order to facilitate multi-disciplinary design solutions.
C3.2	Develop appropriate skills. Development of appropriate skills in human-centred staff either by training or by job experience.
C3.3	Deploy appropriate staff. Skilled staff are involved and effective in all stages of development as and when required.

Table I.7: HF Skills Attribute [68]

ID	Practice
D1.1	Integrate HF processes. Integration of quality in use processes with quality system.
D1.2	Facilitate interface between HF and the organisation. Ensure that the department promoting a human-centred approach understands and uses the language and working methods appropriate to successful interaction with other departments.
D1.3	Use appropriate representations. Representations of user requirements and changes to the system arising from user involvement should be understandable by system developers and programmers.

Table I.8: Integration Attribute [68]

ID	Practice
D2.1	Ensure design feedback. Ensure that evaluations take place at all stages in order to influence the system to be delivered.
D2.2	Change based on feedback. The development process encourages design changes based on actual user experience.
D2.3	Timing of feedback. Ensure that information on user needs and quality in use defects is fed into the design process at appropriate times and in the right format for use.

Table I.9: Improvement [68]

Human-centred processes are integrated into the quality process and systems life cycle of the organisation. The systems and human-centred life cycles are managed to ensure that the results of the human-centred processes produce improvements in all relevant work products. The required time and resources are provided for revision to improve quality in use. Information derived from human-centred processes is in a suitable format to be easily assimilated by relevant staff. The following attributes of the process demonstrate the achievement of this level, in addition to the attributes for all the previous levels:

D.1 Integration Attribute

The extent to which Human-centred processes are integrated with other processes. In order to achieve this level of maturity an organisation should carry out the following:

D.2 Improvement Attribute

The extent to which Human-centred processes are used in the improvement of work products from other processes. In order to achieve this level of maturity an organisation should carry out the following:

D.3 Iteration Attribute

The extent to which the development life cycle is iterative. In order to

ID	Practice
D3.1	Minimize risks by iteration of design. Iteration of the design using prototypes etc. increases the match between the final system and user expectations
D3.2	Manage iteration of design solutions. Information should be recorded to manage the progress of iterative design.
D3.3	Use design objectives to control iteration. Manage the prototyping process by setting and monitoring target quality in use levels set for particular aspects of the system.

Table I.10: Iteration Attribute [68]

ID	Practice
E1.1	Manage usability programme. Management of the whole programme of human-centred processes on all projects in a department or organisation.
E1.2	Systematic improvement of quality in use. Use quality in use defects to analyse and improve problems with the organisation's processes, thereby reducing quality in use defects.
E1.3	Human-centred improvement of organisation. The approaches used to ensure that systems are human-centred are also used within the organisation to improve its own processes and systems.

Table I.11: Human-Centred Leadership Attribute [68]

achieve this level of maturity an organisation should carry out the following:

Level E: Institutionalized

The quality in use of whole ranges of systems is coordinated and managed for business benefit. The culture of the organisation gains benefit from being user and human centred. Human-centred processes are used within the organisation to improve the quality in use of the processes, tools and methods used and developed by the organisation for its own use. Quality in use defects are treated on equal terms with other system defects. Human-centred skills are regarded on a par with engineering skills. The following attributes of the process demonstrate the achievement of this level, in addition to the attributes for all the previous levels:

E.1 Human-Centred Leadership Attribute

The extent to which the human factors/people-centred approach influences the management of all systems life cycle processes. In order to achieve this level of maturity an organisation should carry out the following:

E.2 Organisational Human-Centredness Attribute

The extent to which the human factors/people-centred approach influences the attitude of the organisation. In order to achieve this level of maturity an organisation should carry out the following:

ID	Practice
E2.1	Organisational implementation of user-centred practices. Assist the organisation in the establishment and use of human-centred tools and methods, and in maintaining a focus on the consideration of user issues.
E2.2	Acceptance of human-centred skills. Recognition of the pivotal role played by human-centred skills in an integrated development team. Management of the resources available - human and others.

Table I.12: Organisational Human-Centredness Attribute [68]

Appendix J

Glossary

Agile Methodologies: are lightweight methods that tackled the set of perceived limitations of plan-driven methodologies via providing a reasonable compromise between absence of a process and excessive process [88, 105]. Blomkvist [23] describes "Agile" as an umbrella term that embodies a set of different lightweight methods or processes that share core values and principles related to software development process [23].

Capability: Process or practice ability to achieve desired goals [69].

Capability Level: Evolutionary successive stages or levels that signify step by step patterns of evolution and change designating the desirable or current organizational capabilities against a specific class of entities [94, 201, 248]. Those maturity levels form a path from initial state to maturity that can describe logical, anticipated, or desired evolution and change path(s) [16, 94].

Capability Maturity Model: Normative [140] reference models [110] that embrace the assumption of predictable evolution and change patterns. The main purpose of maturity models is to assess the current situation in order to evaluate the strengths and weaknesses and then prioritize and plan for improvement [140]. This is achieved via evolutionary successive maturity stages or levels.

Context of Use: The tasks, users, equipment, and environment of using a system [69].

Contextual Design: Contextual design is a structured approach to gathering and representing fieldwork information in order to utilise it in design endeavours [241].

Discount Usability Inspection: This is an concept coined by Jakob Nielsen to promote design and testing techniques that are low-cost and simple [211, 213] as an alternative to the high cost of traditional usability design and inspection techniques.

Field Studies: Field studies is an example of usability evaluation paradigms that are conducted in a natural settings in order to understand users tasks in their natural context and the effect of technology on them. Then this can be used to explicate the need for new technology via determining design requirements and facilitating the introduction of technology [241].

Focus Groups: Interviews are restricted by one person's perspective at a time. Thus focus groups emerged to offer an alternative to the restriction when there is a need to gather a group of stakeholders together to discuss issues and requirements [163, 241].

Goal: Intended Outcome.

Heuristic Evaluation: A usability engineering technique where user interface is evaluated against known design principles, heuristics. The goal of the evaluation is to find usability problems in the interface.

Interaction Design: Preece et al. [241] declares that Interaction design is achieved via a user centred approach to design. Interaction design involves four iterative core activities: establishing requirements and identifying needs, developing alternative designs, building interactive design versions, and usability evaluation of what is being built throughout the process. The interaction design process has three key characteristics: user involvement throughout the development process, early identification and documentation of user experience and usability goals and iteration through interaction design activities.

Interviews: Interviews are an example of UCD techniques that involve asking someone a series of questions. Interviews tend to be one to one in order to elicit one person's perspective at a time [163, 241]. Interviews can be classified into structured, semi structured or unstructured, depending on how rigourously the interviewer abides by a prepared set of questions [163, 241].

Life Cycle: The development, operation and maintenance of a system spanning from the definition of system requirements to the termination

of system use [69].

Maturity: Process ability to achieve a desired goal [69].

Maturity Model Assessment: Maturity model assessment focus on comprehending and enhancing the process under investigation [106].

Maturity Model Evaluation: Maturity model evaluation focus on understanding and improving the maturity model itself independently of the assessments or based on assessment results [106].

Although, maturity models are perceived as an assessment and improvement tools. Maturity models are also subject to evaluation and improvement activities [106]. *Maturity Model Assessment* focus on comprehending and enhancing the process under investigation whereas the evaluation focus is to understand and improve the maturity model itself. *Maturity Model Evaluation* on the other hand can be conducted independently of the assessments or based on assessment results [106]. Helgesson et al. [106] conducted a systematic mapping study and selected 59 papers relevant to maturity models' evaluation and assessment of maturity models. He then proposed a maturity models evaluation framework [106].

Naturalistic Observations: Humans usually face difficulty in accurately explaining their tasks and how they achieve it. Thus some UCD techniques, for example, questionnaires, interviews, focus groups, thinking aloud can fall short in providing the designer with an accurate and complete picture [163, 241]. Naturalistic observation can complement the above techniques and provide context for tasks via involving a member of the design team to shadow a stakeholder and make notes, ask questions and observe what is performed in the natural activity context [163, 241]. The level of observer's involvement varies between no involvement (outside observation) and full involvement (participant observation) at the other end.

Practice: A measurable process aspect that is related to the process maturity [69]. Processes are achieved through the implementation of a set of practices.

Predictive Evaluations: This is an example of usability evaluation paradigms that utilise experts' knowledge of typical users to foresee usability problems [241]. Experts are usually guided by heuristics or theoretically based models. This process is referred to as discount evaluation since it is relatively inexpensive and quick since it does not require users presence or special facilities [241].

Primary Study: An empirical study investigating a specific research question.

Process: Interrelated set of activities that transform inputs into outputs [69]. Processes and their associated practices utilize and produce associated work products that can be in the form of pieces of information, documents, hardware, software, training courses, awareness in individuals, etc.

Process Assessment: The utilisation of an assessment model in order to evaluate an organization's software processes [69].

Process Capability Determination: An assessment of a chosen software processes against a target capability. The results are used to identify the weaknesses and strengths of the process under evaluation [69].

Process Improvement: The endeavours performed by an organisation to change its processes in order to achieve business needs and goals more effectively [69].

Prototype: Prototypes refers to any artefact created for the purpose of demonstration to users in order to elicit or test user feedback [69]. A test version or a demo of a program. Prototypes can be used for testing and evaluation of a user interface and usability of a program [103]. Representation of all or part of an interactive system,that although limited in some way, can be used for analysis, design and evaluation.

Questionnaires: Questionnaires are an example of UCD technique that represents a set of questions designed to collect specific information and are usually remotely administered; sent electronically or are posted on a website, and sometimes they are delivered on paper [163, 241].

Quick and Dirty Evaluation: This is an example of usability evaluation paradigms, it occurs when designers informally gather feedback from users or consultants on their ideas and can be used throughout the development life cycle when fast input is required [241]. The collected data is usually descriptive and informal and fed back into the design process [241].

Scenario: A scenario is "an informal narrative description" [37]. Scenarios describe user tasks and activities in a story form thus allowing the exploration and discussion of requirements, contexts, and needs [241].

Scenario Based Design: Scenario based design revolves around combining scenarios usage with claims that expresses the negative and positive effects of specific design features [38, 250]. Scenarios utilise the vocabulary

and phrasing of user thus they are easy to understand by stakeholders and form a powerful mechanism for communicating among team members and with users and stakeholders are usually actively involved in producing and validating scenarios [241].

Software: A generic term for programs running on a computer. Software consists of instructions given in a specific programming language for a computer to perform certain operations [103].

Software Developer/Software Engineer: A professional with usually technical background and/or education who is responsible for the technical design and implementation of software products or systems. Both terms are used interchangeably [103].

Software Development Methodology (SDM): SDM is defined as "A recommended collection of phases, procedures, rules, techniques, tools, documentation, management and training used to develop a system" [10].

Systematic Literature Review: A form of secondary study that utilises well defined methodology to identify, analyse and interpret all available evidence relevant to a specific research question in an unbiased and repeatable manner.

Systematic Review Protocol: A plan that describes the conduction of a proposed systematic literature review.

Task: A task is a set of discrete actions taken to reach certain goals [103].

Task Analysis: Task analysis is the process of investigating physical actions and cognitive processes [241].

Thinking Aloud: A UCD technique for examining individuals' problem solving strategies that requires participants to externalize their thoughts by speaking out loudly everything that they are thinking of and attempting to do [163, 241]. The evaluator is responsible of interrupting the participant in case he falls into silence and reminding him to think out loud [163, 241]. This technique was found to be successful with children and when evaluating systems intended to be used synchronously by groups of users [241].

Usability: The usability of a product is the consequence of systematic user centred design work that occurs throughout the development process and continues even after product release in order to enhance subsequent versions [59, 97]. Usability is also defined as "the capability of the software product to be understood, learned, used and attractive to the user, when

used under specific conditions" [137] as cited in [259]. Another definition by ISO/DIS9241-11 [138] defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" as cited in [259]. In addition, IEEE-Standard-1061 [134] defines usability as "the ease with which a user can learn to operate, prepare inputs for and interpret outputs of a system or component" as cited in [259]. Jokela and Abrahamsson [152] defines usability as "a quality attribute of a product that is dependent on the extent and performance of user centred design activities in a specific development project". Finally, Nielsen and Phillips [216] defines usability as "The absence of obstacles that prevent users from completing their tasks with the system."

Usability Engineering: Usability engineering involves specifying formal, verifiable and quantifiable usability criteria [214]. This is achieved via determining measurable criteria for product performance and assessing the product against these measures then utilising the assessment results in making changes to subsequent system versions [241].

Usability Engineering Life Cycle: Usability engineering life cycle provides a holistic view of usability engineering and a detailed description of the methods to execute usability tasks and integrate it into traditional software development life cycles [191]. The usability engineering life cycle has essentially three tasks: requirements analysis, design/testing/development, and installation [241].

Usability Testing: Usability testing is an example of usability evaluation paradigms that involves measuring typical users' performance on carefully prepared system tasks while watching and recording users performance and logging their software interactions [241]. This observational data is used to calculate performance times, compute time of task completion, identify number and type of errors and explain reasons behind users actions [241].

User: Anyone who employes a system or an artefact to achieve a task [69]. A person who uses a program or interacts with a computer system. Users are always natural human beings with personal differences, goals, and needs that should be considered in software development. Synonym to the term end user is also found in literature [103].

User Centred Design: The usability of a product is the consequence of systematic user centred design work that occurs throughout the development process and continues even after product release in order to enhance

subsequent versions [59, 97]. UCD goal is to satisfy users via producing usable and understandable products that meet their needs and interests [59, 219, 241] in addition to their goals, context of use, abilities and limitations [23]. This goal is accomplished via a set of techniques, methods, procedures and processes as well as a philosophy that places the user at the centre of the development process in a meaningful, appropriate and rigourous ways [39, 59, 95, 219, 241].

User Centred Design Practitioner: A professional, regardless of his or her education or job description, who is experienced in usability engineering and is responsible for good user interface design, practicing of usability methods, and the implementation of user centred design in the organization [103]. This term is used instead of the following terms: UCD specialist, usability specialist, interface designer, designer, interaction designer, usability engineer, user experience designer, usability interface engineer, user experience engineer, etc.

User Experience: The (ISO standard 9241-210, 2008) defines user experience as "A person's perceptions and responses that result from the use or anticipated use of a product, system or service." Garrett [92] defines user experience as the experience that users go through via using the product. User experience focus is on how a user perceives a product that he works with. User experience is perceived as a critical criteria in differentiating between a successful and unsuccessful product [92].

User Interface: All components of an interactive system (software or hardware) that provide information and controls for the user to accomplish specific tasks with the interactive system.

Appendix K

A Maturity Model for Integrating Agile Development Processes and User Centered Design

Developed By

DINA SALAH

Supervised By

DR.RICHARD PAIGE AND DR.PAUL CAIRNS

UNIVERSITY OF YORK

COMPUTER SCIENCE

June 2013

Declaration

This maturity model documentation has been created after the thesis was examined, in order to provide an independent reference. While the documentation is officially dated 2013, this is the date of the thesis examination; the documentation itself was created after the examination, in 2014.

K.1 Abstract

This documentation describes a lightweight, descriptive maturity model for integrating Agile development processes and User Centred Design (UCD). The model is up to our knowledge the first maturity model to approach the process of Agile and User Centred Design Integration (AUCDI) maturation and its constituents. This model contributes to provision of structure of Agile and User Centred Design Integration (AUCDI) efforts via providing organisations with a means for both process definition and process assessment.

The maturity model is composed of three components: first, a multidimensional reference model that includes a set of fundamental elements that affect AUCDI and thus should be be reflected in the model and examined in an assessment. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. Third, an assessment procedure to provide practical guidance for performing the assessment.

Keywords: Capability, capability scale/level, capability maturity model, maturity, practice, practice rating, process, process assessment, process capability determination, context of use, user experience, user interface, process improvement, stakeholder, software, software developer/software engineer, task, task analysis, usability, usability testing, user, user centred design, user centred design practitioner, work products.

K.2 Executive Summary

This documentation describes a lightweight, descriptive maturity model for integrating Agile development processes and User Centred Design (UCD). This model contributes to provision of structure of Agile and User Centred Design Integration (AUCDI) efforts via providing organisations with a means for both process definition and process assessment.

AUCDI Maturity Model Components

The AUCDI maturity model is composed of three components as follows:

1. **A Multidimensional Reference Model:** This includes a set of fundamental elements that affect the integration of Agile development processes and user centred design and thus should be examined in an assessment. These dimensions are: UCD infrastructure; AUCDI process; people; and UCD continuous improvement.
2. **Performance Scale (Scoring Scheme):** This scale rates the project's and organisation's performance in the assessed elements included in the AUCDI reference model.
3. **AUCDI Assessment Procedure:** This procedure provides guidance to AUCDI assessors in their endeavour to assess the organization's and project's capability to integrate Agile developement processes and user centred design.

Capability Levels

The AUCDI maturity model includes six levels of capability as follows:

Level 0: Not Possible. AUCDI is discouraged.

Level 1: Possible. AUCDI is not discouraged.

Level 2: Encouraged. Organisational culture encourages AUCDI.

Level 3: Enabled/Practiced. AUCDI is practiced.

Level 4: Managed. Employees are expected to perform AUCDI.

Level 5: Continuous Improvement. AUCDI processes are reviewed for assessing the status-quo and plan for improvement.

Further details on the development of the AUCDI maturity model, maturity levels, components, and evaluation is provided in this document.

Approval

Author: Dina Salah

Feedback

This document is developed by the University of York, Computer Science Department. Any feedback on this document and its contents should be directed to Dina Salah at the University of York.

Dina Salah

Research Student

Department of Computer Science

University of York.

Deramore Lane. York. YO10 5GH

email: dm560@york.ac.uk

Tel: (+44 (0)1904 325609)

K.3 Glossary of Terms

The following terms are related to the domain of integrating Agile development processes and user centred design, usability assessment, usability capability or usability maturity and are necessary for understanding this documentation.

Capability: Process or practice ability to achieve desired goals [69].

Capability Scale/Level: Evolutionary successive stages or levels that signify step by step patterns of evolution and change designating the desirable or current organizational capabilities against a specific class of entities [201, 248]. Those maturity levels form a path from initial state to maturity that can describe logical, anticipated, or desired evolution and change path(s) [16].

Capability Maturity Model/Maturity Model: Normative [140] reference models [110] that embrace the assumption of predictable evolution and change patterns. The main purpose of maturity models is to assess the current situation in order to evaluate the strengths and weaknesses and then prioritize and plan for improvement [140]. This is achieved via evolutionary successive maturity stages or levels.

Context of Use: The tasks, users, equipment, and environment of using a system [69].

Maturity: Process ability to achieve a desired goal [69].

Practice: A measurable process aspect that is related to the process maturity [69]. Processes are achieved through the implementation of a set of practices.

Process: Interrelated set of activities that transform inputs into outputs [69]. Processes and their associated practices utilize and produce associated work products that can be in the form of pieces of information, documents, hardware, software, training courses, awareness in individuals, etc.

Process Assessment: The utilisation of an assessment model in order to evaluate an organization's software processes [69].

Practice Rating: The utilisation of an assessment model in order to evaluate a particular practice.

Process Capability Determination: An assessment of a chosen software processes against a target capability. The results are used to identify the weaknesses and strengths of the process under evaluation [69].

Process Improvement: The endeavours performed by an organisation to change its processes in order to achieve business needs and goals more effectively [69].

Software: A generic term for programs running on a computer. Software consists of instructions given in a specific programming language for a computer to perform certain operations [103].

Software Developer/Software Engineer: A professional with usually technical background and/or education who is responsible for the technical design and implementation of software products or systems. Both terms are used interchangeably [103].

Task: A task is a set of discrete actions taken to reach certain goals [103].

Task Analysis: Task analysis is the process of investigating physical actions and cognitive processes [241].

Usability: The ISO Standard (1996) defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" as cited in [259]. In addition, the IEEE Standard in 1998 defines usability as "the ease with which a user can learn to operate, prepare inputs for and interpret outputs of a system or component" as cited in [259].

Usability Capability: A characteristic that determines organisational ability to develop products competitive and high usability level [152].

Usability Capability Maturity Models (UCMMs): UCMMs aim to assist organizations in conducting a systematic analysis that evaluates the strengths and weaknesses of the organization in regards to UCD related aspects [153] and accordingly plan for improvement actions [152].

Usability Testing: Usability testing occurs via assessing users' performance on carefully prepared system tasks while watching and recording users' performance and logging their software interactions [241]. This observational data is used to calculate performance times, compute time of task completion, identify number and type of errors and explain reasons behind users actions [241].

User: A person who uses a program or interacts with a computer system. Users have personal differences, goals, and needs that should be considered in software development. Synonym to the term end user is also found in literature [103].

User Centred Design: A product usability is the consequence of systematic user centred design work that occurs throughout the development process and continues even after product release in order to enhance subsequent versions[59, 97]. UCD goal is to satisfy users via producing usable and understandable products that meet their needs and interests [59, 241] in addition to their goals, context of use, abilities and limitations [23]. This goal is accomplished via a set of techniques, methods, procedures and processes as well as a philosophy that makes the user the center of the development process in a meaningful, appropriate and

rigorous ways [39, 59, 95, 241].

User Centred Design Practitioner: A professional, regardless of his or her education and/or job description, who is experienced in user centred design and is responsible for elicitation of user and user interface design requirements, determining user groups, performing task analysis and user analysis, designing user interaction, designing user interface and conducting usability evaluations. The term is used as a generic term to cover the following terms: UCD specialist, usability specialist, usability engineer, designer, or interaction designer.

User Experience: The (ISO standard 9241-210, 2008) defines user experience as "A person's perceptions and responses that result from the use or anticipated use of a product, system or service."User experience is defined as "the experience that users go through via using the product"[92].

User Interface: The software and hardware components of an interactive system that allow the user to achieve his tasks.

K.4 Introduction

This section provides a foundation for important concepts relevant to integrating Agile development processes and user centred design. This foundation is essential for comprehending the Agile and User Centred Design Maturity Model. It starts by introducing Agile Methods and user centred design and their relevant concepts. Then it highlights maturity models in general and Usability Maturity Models (UMMs) in specific.

Agile methods are lightweight methods that deal plan-driven methods limitations through compromising the absence of a process and excessive process [88]. Agile processes aim to deal with volatile requirements via discarding upfront, precisely defined plans. They are iterative and are used to develop software incrementally. Different Agile processes implement these ideas in different ways. All Agile processes share common values and principles, defined in the *Agile Manifesto* [14].

User Experience (UX) is defined as the perceptions and responses of users that result from their experience of using a product [92]. User Centred Design is a set of techniques, methods, procedures and processes as well as a philosophy that places the user at the centre of the development process [59, 95]. The goal of applying UCD is to attempt to satisfy users via producing usable and understandable products that meet their needs and interests [59].

Maturity Models are normative [140] reference models [110] that embrace the assumption of predictable evolution and change patterns. The main purpose of maturity models is to assess the current situation in order to evaluate the strengths and weaknesses and then prioritize and plan for improvement [140]. This is achieved via evolutionary successive stages or levels that signify step by step patterns of evolution and change designating the desirable or current organisational capabilities against a specific class of entities [201, 248]. Those maturity levels form a path from initial state to maturity that can describe logical, anticipated, or desired evolution and change path(s) [16].

Usability Maturity Models (UMMs) or Usability Capability Assessment Models (UCAMs) are methods for developing UCD processes in companies in order to facilitate usability methodologies for creating usable products [151]. Usability maturity models aim to assist organisations in conducting a systematic analysis that evaluates the strengths and weaknesses of the organisation in regards to UCD related aspects [153] and accordingly plan for improvement actions [152]. Usability Capability is defined as

> A characteristic of a development organisation that determines its ability to consistently develop products with high and competitive level of usability [152].

K.5 Rationale of the AUCDI Maturity Model

Agile and User Centred Design Integration (AUCDI) gained increased interest due to three reasons: first, the reported advantages of UCD on the developed software as it enables developers to understand the needs of the potential users of their software, and how their goals and activities can be best supported by the software thus leading to improved usability and user satisfaction. Second, none of the major Agile processes explicitly include guidance for how to develop usable software [178]. The interaction design role, usability, and UI design in an Agile team is also unclear and largely overlooked [20]. Furthermore, principles and practices for understanding and eliciting usability and user requirements and evaluating Agile systems for usability and UX are generally considerably deficient [156, 178]. Third, there exists philosophical and principled differences between Agile methods and UCD in focus, evaluation method, culture and documentation that suggest that their integration will be fundamentally challenging.

Improving user centred design effectiveness in software development is a considerable challenge in many organizations. Usability maturity models aim to assist organizations in conducting a systematic, status-quo analysis that evaluates the weaknesses and strengths of the organisation in regards to UCD related aspects [153] and accordingly plan for improvement actions [152].

Although AUCDI research started in 2001 and is still growing, nevertheless it has not exploited the research potential on usability maturity models in general and Agile and user centred design maturity models in particular. Until now, the process of becoming more AUCDI mature has not been directly approached. Moreover, the process of AUCDI maturation and its constituents has not been tackled/ approached. As a result there is no research available on what constitutes AUCDI maturity and the process of AUCDI maturation. There is an absence of an AUCDI maturity model that provides organizations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI and that provides organizations with a diagnostic tool to assess both the capability and performance for AUCDI in order to pinpoint its weaknesses and strengths in deploying Agile processes and UCD, determine whether the organization is sufficiently mature for AUCDI and identify the potential difficulties/challenges that could develop during the AUCDI process in order to mitigate them beforehand.

As a result the Agile and User Centred Design Integration Maturity Model was developed to fill this research gap via proposing a lightweight, descriptive multidimensional maturity model for integrating Agile development processes and user centred design. The AUCDI maturity model can be used by organizations for two purposes: AUCDI process definition and assessment as it will be illustrated in section K.6.

K.6 Purpose of Use

The AUCDI maturity model can be used by organizations for two purposes: AUCDI process definition and assessment.

AUCDI Process Definition

Process definition is embodied via providing organisations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI. This AUCDI maturity model provides organisations with a profound and thorough understanding of AUCDI specifics, activities, roles, timing, responsibilities, success factors, and challenges.

AUCDI Process Assessment

Process assessment focuses on providing organizations with a diagnostic tool to assess both the capability and performance for AUCDI. Capability assessment falls on the organization as a whole and it attempts to identify the extent to which UCD is consistently and systematically implemented in the different projects. Whereas, performance assessment focuses on effective UCD implementation at individual development projects level. The process assessment results in identifying the AUCDI weaknesses and strengths. The results of this assessment can be communicated to: first, management to provide them with a better understanding of the issues involved in consistently developing products with high and competitive usability level as well as pinpoint AUCDI hindrance. Second, developers to provide them with a better understanding of usability and UCD. Third, UCD practitioners by pinpointing areas that require improvement in usability processes and practices.

K.7 Knowledge Base (Basis of the Model)

A variety of sources were used as the knowledge base for constructing the AUCDI dimensions. This knowledge base is explained in full details in Dina Salah PhD Thesis, Department of Computer Science, University of York. This documentation only includes a brief about these sources. The knowledge base of the AUCDI maturity model depended on the following sources:

Theoretical Sources: The first theoretical source was the Agile Manifesto [14] so as to ensure that none of the proposed processes or practices for integration conflict with Agile values and principles. Second, a number of sources for describing UCD principles and activities so as to ensure concrete guidance in regards to integrating UCD into the overall project plan and all phases in the product development life cycle via clear milestones for UCD activities along the software development process. Thus UCD principles and activities discussed in ISO 13407, and UCD Processes from KESSU 2.2 [146] were taken into consideration in developing the AUCDI maturity model. Third, principles of development and design of maturity models that were proposed in [16, 57, 201, 240].

Literature Reviews: The proposed AUCDI maturity model took into consideration a number of issues: first, Agile and UCD differences and commonalities since they represent divergence and convergence points that can hinder or enhance the integration. Second, AUCDI integration success factors in order to include them as integration processes or practices in the proposed AUCDI dimensions. Thus two literature reviews were conducted, the first, a systematic literature review that provided the differences and commonalities between Agile and UCD, AUCDI challenges, success factors and practices. The second literature review focused on usability maturity models. The results of the UMM literature review revealed the deficiencies of the usability maturity models that are available in the public domain in regards to: the quantity of published research in the public domain in general and on empirical validation in particular and lack of a UMM that is initially created for use in the context of Agile development processes.

Empirical Sources: Two empirical studies were conducted and their findings, observations and lessons learned provided insights for the proposed AUCDI maturity model. These empirical studies involved: first, 14 interviews of AUCDI industrial attempts that provided insights in regards to AUCDI difficulties and practices utilised by industrial practitioners to tackle these difficulties. Second, an interview study that utilised Nielsen Capability Maturity Model and UMM-HCS in assessing the usability maturity level of five case studies that integrated Agile development processes and user centred design. The findings, observations and lessons learned from Nielsen model and UMM-HCS revealed their deficiencies in addressing the specifics, requirements, activities, success factors and challenges identified within the AUCDI domain.

K.8 Maturity Stages for Agile and User Centred Design Integration Maturity Model

The maturity stages for AUCDI maturity model can be illustrated from table K.1.

Stage	Definition
Level 0: Not Possible	AUCDI is discouraged. There is a general unwillingness to integrate UCD into the Agile development process. Management and development teams do not seem to value or understand UCD.
Level 1: Possible	AUCDI is not discouraged. There exists a general willingness to perform it. One or few development team members understand the value and meaning of UCD and the benefits of AUCDI.
Level 2: Encouraged	Organisational culture encourages AUCDI. Value, benefits and meaning of UCD is recognised. AUCDI is achieved in some projects.
Level 3: Enabled/Practiced	AUCDI is practiced. UCD methods, tools, workspace and qualified staff exist to enable the integration activities. Management supports and promotes the integration of Agile and UCD.
Level 4: Managed	Employees are expected to perform AUCDI. Training is available. AUCDI activities are part of the software development life cycle. UCD methods, tools, workspace, cycles and qualified staff for supporting AUCDI activities are available. Team Leaders (Agile coaches or Scrum masters and product owners) exhibits awareness and commitment to UCD and provides a strategy for achieving Agile UCD integration throughout all development projects. Development team exhibits awareness and commitment to AUCDI. Customer(s) exhibits awareness and commitment to AUCDI.
Level 5: Continuous Improvement	AUCDI processes are reviewed for assessing the status-quo and plan for improvement. UCD methods, tools, guidelines, workspace and qualified staff are widely accepted and regularly monitored and continuously improved.

Table K.1: Maturity Stages for Agile User Centred Design Integration Maturity Model

K.9 AUCDI Maturity Model Components

The AUCDI maturity model is composed of three components: first, a multidimensional reference model that includes a set of fundamental elements that affect AUCDI and thus should be be reflected in the model and examined in an assessment. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. Third, an assessment procedure to provide practical guidance for performing the assessment. Further details on these components will be included in the following sections:

K.9.1 Component 1: Multidimensional AUCDI Reference Model

The first component of the AUCDI maturity model is the multidimensional AUCDI reference model. This reference model is composed of a set of dimensions that represent fundamental elements that affect the integration of Agile development processes and UCD and thus should be examined in an assessment. These elements are included in the AUCDI reference model. The results of the assessment can help organisations assess their current status and all the factors that impact AUCDI process and pinpoint weaknesses and strengths in order to pinpoint improvement areas.

A domain component is a major, independent aspect that is significant to a particular domain maturity e.g. critical success factors, barriers to entry [57]. Figure K.1 represents domain components/dimensions and sub components identified for the AUCDI maturity model.

The multidimensional AUCDI reference model is composed of four main dimensions: UCD infrastructure, AUCDI process, people and UCD continuous improvement.

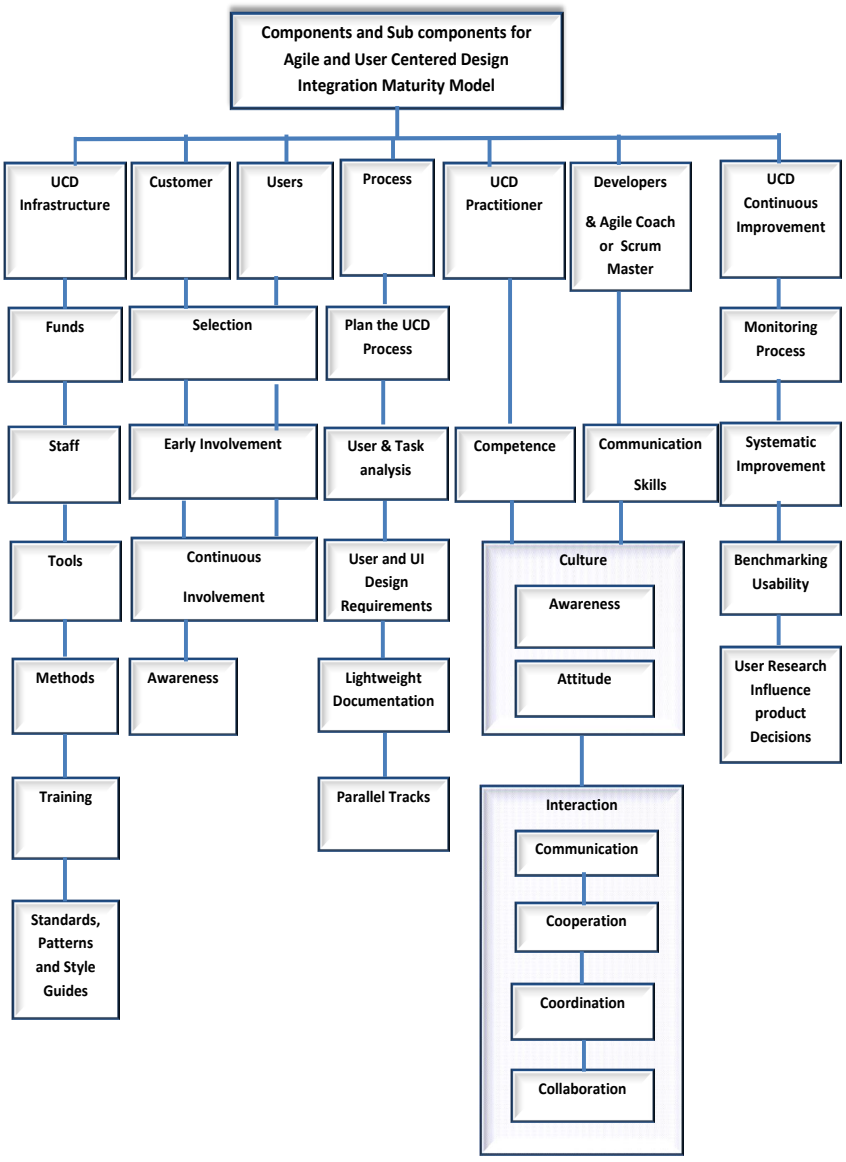

Figure K.1: Domain Components and Sub Components for the Agile User Centred Design Integration Maturity Model

K.9.1.1 Dimension 1: UCD Infrastructure

UCD infrastructure involves a number of organisational elements that need to be available in order to achieve successful integration of Agile development processes and UCD. The importance of UCD infrastructure is that it signifies a high maturity for AUCDI since in the absence of this infrastructure AUCDI will be dependent on the efforts and interest of a usability champion or development team members in achieving the integration. As a result the integration will occur on per project basis based on the availability of development team skills and interest rather than on an organisational policy that encourages and enforces AUCDI to occur throughout all projects. UCD infrastructure as a component is composed of a number of sub components as it will be illustrated below:

Funds: The first component of UCD infrastructure dimension is related to the allocation and utilisation of dedicated funds for conducting UCD related activities. These funds can be used in conducting early user research, field studies, usability tests, setting usability labs, buying prototyping tools, hiring qualified UCD staff, conducting relevant UCD training programs, setting an open space workplace for UCD practitioners and developers, etc.

Staff: The second component of UCD infrastructure dimension is the presence and utilisation of qualified UCD practitioners. Further details on UCD practitioners' qualifications will be provided later in section 8.2.3.3.

Tools: Integrating UCD into Agile development processes requires the presence and utilisation of tools. Examples for these tools are: usability labs and design, prototyping, and documentation tools, etc.

Methods: Appropriate UCD methods should be used in carrying out different UCD activities. These could include user requirements elicitation methods, for example, questionnaires, interviews, focus groups, observations, thinking aloud, etc. Moreover, these also include usability inspection methods, for example, discount usability testing methods, etc.

Management Awareness and Support: In the context of AUCDI, management should be aware of the importance of UCD and the UCD practitioner's role. Management should understand that UCD must be part of the business strategy and support the inclusion and prioritization of UCD activities in the Agile development process.

Training: Achieving AUCDI requires awareness on both Agile and UCD related aspects. Agile awareness on one hand includes Agile values, principles, and practices. Agile awareness training should be provided to a number of parties including: UCD practitioners, customers, and business managers.

UCD awareness includes UCD principles and activities. UCD awareness training

should be provided to a number of parties including customers, business managers, developers, Agile coaches or scrum masters and product owners.

Finally UCD practitioners should receive awareness training about Agile developer's role and Agile developers should receive awareness training on UCD practitioner's role. This awareness training will help in setting expectations of both parties from each other as well as help them synchronize their efforts and achieve productive communication.

Standards, Patterns and Style Guides: Utilisation of standards, patterns and style guides is of special importance to ensure consistency across products and re-usability. In the context of Agile projects this is of special importance since some projects can be developed by small teams and utilization of standards, patterns and style guides help to improve their work. Team formation also varies among different Agile projects and some teams may lack specialized UCD practitioners and as a result UCD work becomes the responsibility of developers. Since developers usually have software engineering education and software development expertise thus they may lack sufficient UCD, usability and user experience knowledge that can be compensated via utilising standards, patterns and style guides.

Colocation of Developers and UCD Practitioners: The systematic literature review conducted suggested that colocation of developers and UCD practitioners in Agile teams offers a number of advantages. These advantages are rapid and instant communication due to the availability of the UCD practitioner to instantly answer developers questions, clarify and address any design issues that emerge, and exchange design in a constant and an ongoing manner. colocation allows developers to influence the design as it progresses and are aware of UCD practitioners' work and can discuss early on any design areas that could have a negative impact on implementation. Thus colocation allows for continuous communication, negotiation, knowledge sharing, instant decision making and improved sense of team.

Nevertheless, the main importance and impact of colocation is that it allows for rapid and continuous communication. Thus in case of the presence of organisational policies, work situations or team commitments that prevent colocation this can be mitigated via introducing a clear process for information sharing, collocating teams for cycle planning meetings, continuous synchronous and asynchronous communication using telecommunication tools, for example, chat, instant messaging, emails, phone, video conferencing, etc.

K.9.1.2 Dimension 2: AUCDI Process

Throughout this section the word *cycle* will be used to refer to both iterations and sprints.

Although UCD infrastructure lays the necessary foundation for integrating Agile development processes and UCD, the success of AUCDI is dependent on a number of other dimensions. The second dimension is the presence of an AUCDI process that takes into consideration the iterative and incremental nature of the Agile development process. This AUCDI process should focus on the planning and implementation of UCD activities and principles into the Agile development life in order to achieve the integration. This involves a number of issues like planning for the inclusion of UCD activities in the project plan, providing both developers and UCD practitioners with a road map on their roles and responsibilities, executing UCD activities throughout the Agile cycles and synchronizing the efforts of UCD practitioners and developers, etc.

The proposed AUCDI process is a method independent process that seamlessly integrates AUCDI related activities throughout the Agile development life cycle. The AUCDI process integrates UCD into the overall project plan and all phases in the product development life cycle via clear milestones for UCD activities along the software development process. This is achieved via including detailed activities of user requirement gathering, feedback and design evaluation as well as explicating integration work products, work flows, roles, and responsibilities. It is based on the ISO 13407 UCD activities and UCD processes of KESSU 2.2 [147]. However, the AUCDI process extends those founding UCD processes and activities by addressing the specifics, requirements, activities, success factors, and challenges identified within the AUCDI domain.

The AUCDI process adopts parallel tracks [203] and suggests a number of processes for AUCDI that are achieved through the implementation of a set of practices that describe the activities to be accomplished in order to achieve the process. Examples of processes are planning the UCD process, user analysis, task analysis, identification and understanding of user requirements, identification and understanding of UI design requirements, lightweight documentation, synchronization efforts between UCD practitioners and developers, coordination and effective scheduling of UCD practitioners and developers activities, interaction design, user task design, usability evaluation. Each of these processes include a set of subsequent practices.

Processes and their associated practices utilise and produce work products that takes different forms, for example: designs, prototypes, training courses, code, etc.

The AUCDI process is included in figure K.2. This figure include only the main

AUCDI processes and practices. Figure K.2 uses ellipses to refer to processes and squares to refer to outputs of those processes. Further details on the AUCDI process is included in section K.11.

The AUCDI process can be divided into early work performed in cycle N-1, then work performed in cycle N, N+1, N+2, etc. as it will be illustrated in this section that details the essential processes and practices that are related to the AUCDI process.

Early Work-Cycle N-1

Early work performed in cycle N-1 is often referred to in the AUCDI domain as "upfront design". Upfront design is a a separate pre-development period that is used for eliciting user requirements, understanding users, user goals and context of use, utilising backlog items to create initial user stories and conducting design up front and ahead of developers in order to achieve a comprehensive system view. Some teams use this phase to check the technical feasibility of the software via communication between UCD practitioners and architects. It can also be utilised in acquiring feedback from management and sales department. It can also be used by the development and quality assurance teams to work on back end features such as selecting the development environment and system platforms or developing features that has a high cost of development and a low cost of design.

The early work that is performed in the upfront design phase or cycle N-1 is divided into a number of processes as it will be explained below:

Planning the UCD Process

This process is focused on planning the UCD process and it involves a number of practices like identifying and planning customer and user involvement, selecting user centred design methods that will be utilised in the different stages of the Agile development life cycle and identifying the relevant UCD specialist skills required and planning to provide them.

Performing User and Task Analysis

This process is focused on user and task analysis and it involves a number of practices like identification and understanding of user groups, understanding and specifying the context of use. This process also involves conducting task analysis where the non-functional tasks' attributes, problems and risks that users meet when performing tasks are identified and understood.

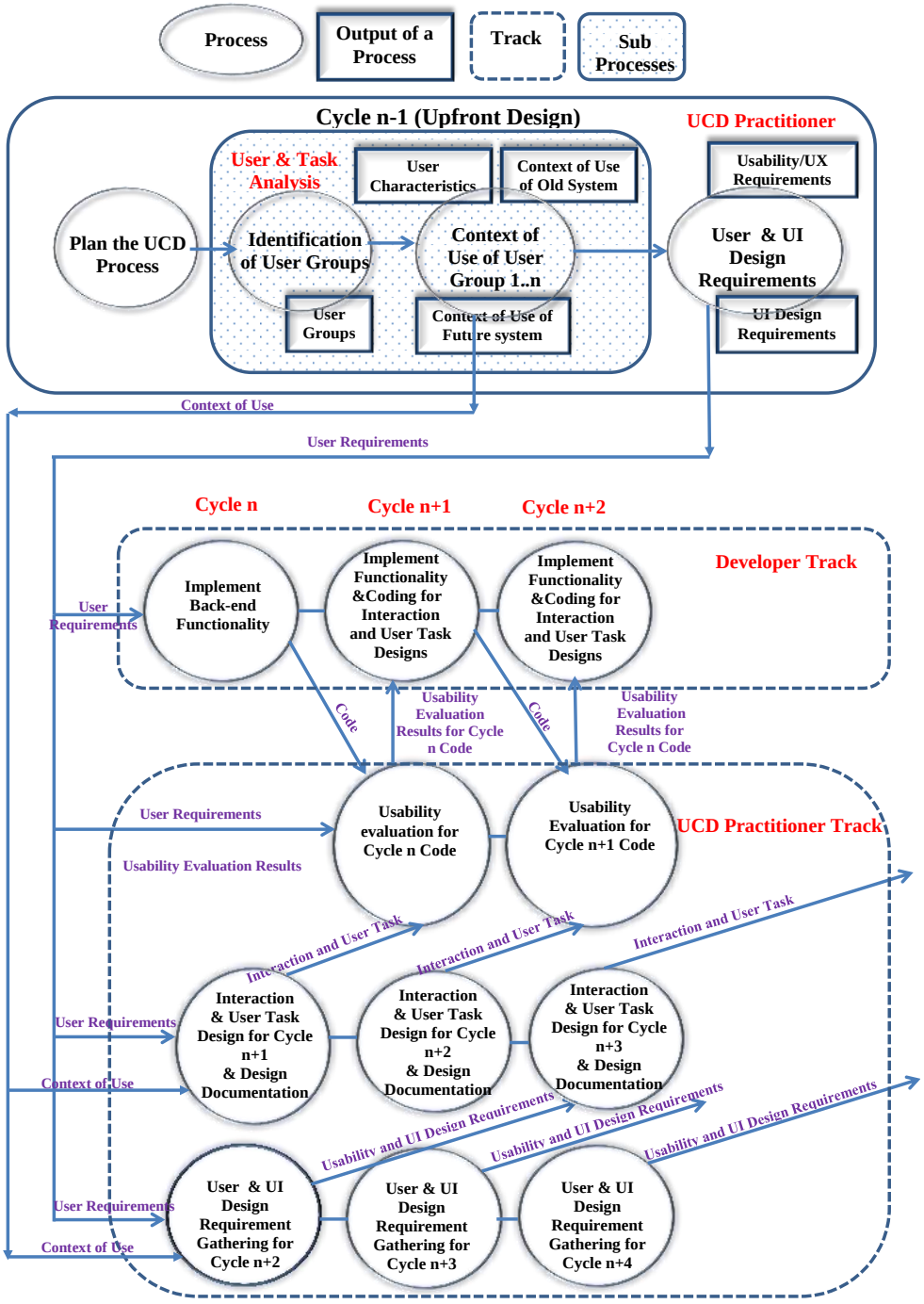

Figure K.2: Agile and User Centred Design Integration Process

Identification and Understanding of User and UI Design Requirements

This process includes two practices that are focused on identification and understanding of user and UI design requirements.

Lightweight Documentation

This process focuses on maintaining lightweight documentation for the results of UCD process planning, user analysis and task analysis, user requirements and UI design requirements, current designs and their expected delivery date, and results of usability tests.

It is important to note that documentation is a recurring task that occurs as early as cycle N-1 throughout N, N+1, N+2, etc.

Cycle Specific Work- Cycle N

Cycle N signifies the starting point for the parallel tracks [203] where the implementation and design are organised as two highly interrelated and equal tracks for effective scheduling and coordination of UCD practitioners and developers activities.

The parallel track involves a cycle 1, where the UCD practitioners work on features that are planned to be implemented in cycle 2. The interface is designed and the testable low fidelity or high fidelity prototype is built. Cycle 1 is also used to rectify and retest design problems that are discovered via usability tests until the designs achieve their design goals. Cycle 1 usually involves developers working on features with high cost of development and little user interface, while the interaction designers work on investigating, creating and verifying next cycles' designs [203]. The parallel track offers a number of advantages for both interaction designers and developers. UCD practitioners on one hand design for the next iteration thus no time is wasted in creating unusable designs and timely feedback on designs is received from developers. Developers on the other hand maximize time for coding since they do not have to wait for the completion of paper prototypes and usability tests by UCD practitioners [203].

This cycle involves a number of processes and practices as follows:

Synchronizing activities for UCD practitioners and developers

UI consistency can be undermined when code is evolved in parallel by independently empowered teams who lack coordination [59]. It was also observed that 92% of the collaborative events between designers and developers were focused on realigning individual work or individual understandings of project and product aims [34]. As a result, the proposed AUCDI process contains a number of synchronization points to allow for close collaboration between UCD practitioners and developers in order to keep the flow of information among all involved

parties. The following practices are all used as synchronization points between the UCD practitioners and developers throughout sprint N.

- **Developers share product vision with UCD team**

  This practice aims to start sprint N by developers sharing the product vision with UCD team in order for UCD practitioners to understand product vision and cooperate with developers in achieving it.

- **UCD practitioner shares user information with the development team**

  This practice aims to allow development team to better understand the user, this will be achieved through the UCD practitioners sharing the results of user analysis, user requirements, UI design requirements with the development team.

- **UCD practitioner shares task analysis information with the development team**

  The aim of this practice is to ensure that developers understand users' needs via understanding the non-functional tasks attributes, problems and risks that users meet when performing tasks.

- **UCD practitioner shares UCD vision with the development team**

  The aim of this practice is to ensure that developers understand the design intent and rationale embraced by UCD practitioners. This lead to having a shared design vision. This visibility of design vision offers a number of advantages including: creating common understanding of important features to the customer, easier prioritization of features [178], minimizing rework, early illumination of integration [277], ease of making decision in regards to competing concerns between developers and UCD practitioners [178] and more effective development of usable software as a result of active participation of stakeholders that results in improved collaboration [199].

Design chunking

The aim of this practice is to break design into cycle sized pieces called design chunks that adds incrementally to the overall design and gradually leads to achieving design goals [272].

Inclusion of UCD activities (usability/user experience features) in product backlog or user stories

The aim of this practice is to ensure the inclusion of UCD activities in product backlog or user stories. The inclusion of UCD activities in product backlog or user stories requires negotiation with developers and Agile coach or scrum master and product owner as well as awareness of UCD importance by developers, Agile coach or scrum master and product owner.

Prioritization of UCD activities/features in cycles

The aim of this practice is to ensure the prioritization of UCD activities/ features in the development cycles. For UCD activities to be prioritized in different cycles this requires both interference from the designers, UCD practitioners or usability product owner as well as awareness of UCD importance by developers, Agile coach or scrum master and product owner.

Developers implementing back-end functionality

The development and quality assurance teams can utilise sprint N in focusing on back end features such as selecting development environment and system platforms or features that are high on costs of development and low on costs of design.

UCD practitioners performing interaction and user task design for cycle N+1.

UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+1.

UCD practitioners gathering user and UI design requirements for cycle N+2

Cycle Specific Work-Cycle N+1

Cycle N+1 is composed of a number of practices as follows:

Developers implementing functionality and coding for interaction and user task designs

UCD practitioner continually clearing up any design questions posed by developers

This practice represents a synchronization point that is used to ensure that developers understand design

UCD practitioners performing usability evaluation for cycle N code

UCD practitioners sharing usability evaluation results for cycle N code with the development team

The aim of this practice is to provide a synchronization point between UCD practitioners and developers. Sharing research and usability findings with the development team allows the team to discuss usability as well as functionality, gives the team a greater sense of accomplishment, demonstrates to stakeholders a collective concern for user satisfaction and quality [136, 209].

UCD practitioners performing interaction and user task design for cycle N+2

UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+2

UCD practitioners gathering user and UI design requirements for cycle N+3

Cycle Specific-Work N+2

Cycle N+2 is composed of a number of practices including the following:

Developers implementing functionality and coding for interaction and user task designs

UCD practitioner continually clearing up any design questions posed by developers

This practice represents a synchronization point that is used to ensure that developers understand design. **UCD practitioners performing usability evaluation for cycle N+1 code**

UCD practitioners sharing usability evaluation results for cycle N+1 code with the development team

This practice represents a synchronization point between UCD practitioners and developers.

UCD practitioners performing interaction and user task design for cycle N+3

UCD practitioners iteratively redoing interaction and user task design for cycle N code according to usability evaluation results acquired in cycle N+1

UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+3

UCD practitioners gathering user and UI design requirements for cycle N+4

K.9.1.3 Dimension 3: People

The third dimension that contributes to AUCDI is the people involved in the process. The AUCDI process involves a number of people including customers, users, developers, UCD practitioners and XP coach in case of utilising XP and program manager and Scrum master and product owner in case of utilising Scrum.

Customers

Agile approaches require development teams to include customer representatives [14]. A number of issues needs to be available in order to achieve optimal customer collaboration in teams striving to achieve AUCDI.

Identification and Selection of Customers: This practice aims to identify and select customers since it was reported that a weak Agile customer can lead to lack of end user participation and making non informed decisions [204]. It was also suggested that an experienced on site XP customer can fill in the technical gap between developers and UCD practitioners [129]. The important issues that are related to customers in order to achieve AUCDI are: awareness, attitude, and continuous involvement.

Customer Awareness of Agile and UCD: This practice aims to ensure that customers are aware of Agile values, principles and practices as well as UCD principles and activities. Understanding Agile on one hand will result in acknowledging the iterative and incremental nature of the Agile development process and what it implies in terms of the time and frequency and form of communication (i.e. face to face). Understanding UCD on the other hand will result in acknowledging the importance of UCD practitioner's role and the needs and expectations of UCD practitioners from the customer. This will result in understanding and fulfilling the needs and expectations of both the UCD practitioners and developers.

Early Involvement: This practice aims to ensure the early focus on customer needs and thus it is preferable for the customers to be involved from the beginning of the project.

Continuous Involvement: This practice aims to ensure continuous customer involvement since the rapid nature of the Agile development process requires the customer to be continuously involved with the development team in order to answer any questions they may have about the product and also to be involved in usability testing.

Users

Software engineering emphasizes the importance of identifying users, understanding their goals and priorities and actively involving them in uncovering requirements [167]. However, as much as it is important to understand the users it is also important to involve uses effectively in the development process and this entails early and continuous involvement of the right users who correctly represent the larger user population [261].

Thus the important practices that are related to users in order to achieve successful Agile and UCD integration are as follows:

Identification and Selection of Users: This practice aims to select the right users who accurately represent the larger user population and involve them in the development process.

Early User Involvement: This practice aims to involve users, who accurately represent the larger user population, in the development process as early as possible.

Continuous User Involvement: This practice aims to ensure continuous user involvement. The involvement of user representatives throughout the whole project results in understanding the project and becoming committed to its purpose [98].

UCD Practitioners

Agile teams' structure varies; some teams have a dedicated UCD practitioner whereas others do not. This can negatively impact the quality of product's usability or user experience. The important features that are related to UCD practitioners in order to achieve successful Agile and UCD integration are: competence, awareness, and attitude.

Competence: This practice aims to ensure competence of UCD practitioners. Competence for UCD practitioners involves knowledge and experience of user centred processes, methods and tools [149] as well as to social skills such as the ability to communicate and cooperate with the rest of the development team.

Awareness: This practice aims to ensure that UCD practitioners are aware of Agile values and principles, what constitutes work for developers, operational challenges and technical challenges and how those challenges impact design [277]. This allows UCD practitioners to set expectations in regards to the time and frequency and form of communication and deliverables with customers, users and developers. It was also reported that when Agile team members perceive the incremental development of user interface as valuable since it resulted in discovering problems early and revealed new requirements. This results in UX practitioners becoming more accepting to Agile development and more cooperative with developers [178].

Attitude: Attitude of UCD practitioner involves a number of issues: continuous communication, respect and trust for developers, and maintaining the visibility of UCD team work.

- **Continuous Communication:** The aim of this practice is to ensure the continuous communication between developers and UI designers in order for designers to answer design questions, receive immediate feedback on design from developers and share usability testing results with developers. This continuous communication will make design decisions more informed and help in avoiding delays or bottle necks in the development process.
- **Respect and Trust for Developers:** The aim of this practice is to optimize communication between developers and UCD practitioners. The existence of trust between designers and developers in each other's professional abilities improves the negotiations of conflicting issues and saves time and effort [82]. UCD practitioners awareness of Agile values, principles and practices as well as what constitutes the role of Agile developers results in increased respect and trust for developers that usually leads to better communication among the Agile team.
- **Maintaining the Visibility of UCD Team Work:** The aim of this practice is to ensure the visibility of UCD team work in order to demonstrate value and promote sustained dialogue. Maintaining visibility of UX team work and making all "work-in-progress" and deliverables highly visible to the entire team results in demonstrating value and promote sustained dialogue [59, 176]. Moreover, the development team should be encouraged to frequently check work in progress; post work flows, wire-frames, screen shots, etc., while allowing them to pose questions, challenge, clarify and propose alternatives [59]. UX team visibility can occur via conveying user research results, designs, user testing results throughout the development process.

Developers

Communication Skills: The aim of this practice is to ensure that developers have communication skills which is essential for group work, specifically in the domain of AUCDI these skills are needed for the developers to continuously communicate their questions, challenges or requests that are related to design and receive UCD practitioners' feedback related to user research, design and user testing results.

Awareness: The aim of this practice is to ensure that developers have awareness of usability and UCD. Developers need to have an awareness in regards to a number of issues including: first, the mechanics of design and how UCD can improve the overall design value and quality [277] or else they may get frustrated due to the time consumed by UCD related activities at early project phases [153]. Second, AUCDI requires mutual awareness between developers and designers in regards to what constitutes work for the other party. This mutual awareness will impact informed judgments in regards to work coordination [86]. Thus development team awareness of UCD enables them to understand the importance of UCD work and the amount of effort involved in user research, design, usability testing, etc and thus have realistic expectations from UCD practitioners. This will result in improved communication and collaboration process and increased respect to UCD practitioners.

Attitude: This practice aims to ensure that developers have the right attitude to ensure the success of the AUCDI process. Attitudes are essential for communication success. In case of people's unwillingness to communicate via listening or sharing their skills and experiences the project will fail regardless of the used tools and methods [99]. The developers' attitude is measured via a number of practices including respect and trust for UCD practitioners, and trust for customer.

- **Respect and Trust for UCD Practitioner:** The aim of this practice is to ensure the developers' respect and trust of developers for UCD practitioner, since it is essential for AUCDI success as it allows better communication, cooperation and collaboration. A number of issues contribute to raising this respect and trust, for example, awareness and visibility of UCD team work.

- **Trust Customer:** The aim of this practice is to ensure that developers trust customers. Trust is important as reported by the results of a focus group in the Agile development conference in June 2003 where participants declared trust as a prerequisite to any act of customer collaboration [261].

K.9.1.4 Dimension 4: UCD Continuous Improvement

This dimension focuses on ensuring the continuous improvement of user centred design. It involves a number of practices as follows:

- Presence of a UCD monitoring process across projects.
- Presence of a systematic improvement process for UCD activities, tools, methods, skills and awareness.
- Benchmarking product's usability and/or user experience against competitive products' usability and/or user experience.
- Product decisions emerge from end user and customer studies and are targeted to meet users needs and expectations.

K.9.2 Component 2: Performance Scale

The second component of the AUCDI maturity model is the performance scale (scoring scheme). This scale helps the assessors to rate organisational performance in regards to the examined AUCDI elements included in the AUCDI reference model. The closer the organisation achieves the AUCDI reference model requirements, the higher its ratings. The scoring scheme chosen aimed to make the scoring for the degree of practice satisfaction accurate and informative. Thus a six point scale was used that included the following ratings: very high (>4.5), high (3.5-4.5), medium (2.5-3.5), and low (1.5-2.5), very low(>0 and <1.5), No (0). This scoring scheme simplifies the computation of the degree of satisfaction of practices and provides easily interpreted names for each of the five degrees of satisfaction and the numerical score provides a finer grain measure. A "don't know" option was also used in case of respondent's inability to answer one or more of the questions due to lack of knowledge or non applicability of the question to the respondent situation.

The AUCDI performance scale is included in section K.10 to differentiate between the different maturity levels.

K.9.3 Component 3: Assessment Procedure

The third component of the AUCDI maturity model is the assessment procedure. The AUCDI assessment procedure provides guidance to AUCDI assessors in their endeavour to assess the organisation's and project's capability to integrate Agile and UCD. The assessment procedure is composed of a maturity recording sheet, maturity levels performance rating, typical quotes, and assessment guidelines as it will be illustrated in the following sections:

Maturity Recording Sheet

The maturity recording sheet is used to provide assessors with a template in which they can record their assessment of the different AUCDI processes and practices. The AUCDI maturity model recording sheet is included in section K.11.

Maturity Levels Performance Rating

The AUCDI maturity levels with their corresponding performance rating are included in section K.10. This can be used later by the assessors to compare the recorded scoring from the maturity recording sheet in section K.11 with the performance rating of the AUCDI maturity levels in order to determine their organisational AUCDI maturity level.

Typical Quotes

A number of typical quotes were put togethere that signify each AUCDI maturity level. Those typical quotes can provide assessors with a benchmark to compare their maturity level with. The AUCDIMM assessor should use these quotes in order to ensure that he has correctly assessed the organisational AUCDI maturity level. The inclusion of typical quotes as a guideline for assessors was used by a number of researchers including [49, 68] in Quality Maturity Grid and UMM-HCS respectively. Section K.12 contains a list of typical quotes per maturity level.

Assessment Guidelines

A set of guidelines for performing the assessment were put together in order to provide assessors by a clear road map for conducting the assessment. Those assessment guidelines are included in section K.13.

Assessment Report

The results of the assessment is utilised in generating an assessment report. This report provides an executive summary including the AUCDI maturity profile and maturity level. The AUCDI maturity profile indicates the degree of satisfaction of each key practice and whether it is unsatisfied, missing or needs improvement.

This section is followed by the references list. It is worth noting that each item included in the reference list is followed by a number or a set of numbers. This signifies the page number(s) where this reference was utilised.

K.10 Agile User Centred Design Integration Maturity Levels

Table K.2 reflects the ratings for the AUCDI maturity levels for the UCD infrastructure as the first dimension of the AUCDI maturity model. The following abbreviations will be used:

- No is abbreviated as N.
- Low is abbreviated as L.
- Very low Low is abbreviated as VL.
- Medium is abbreviated as M.
- High is abbreviated as H.
- Very High is abbreviated as VH.

Item	Practices	ML0	ML1	ML2	ML3	ML4	ML5
Funds	Presence/utilization of dedicated funding for early user focus, field studies, usability and / or user experience activities, etc.	N	VL	L	M	H	VH
Staff	Presence of qualified UCD team led by UCD manager	N	N	N	N	H	VH
	Presence of qualified UCD practitioner per project/in this project	N	N	N	M	H	VH
Tools	Availability and utilization of prototyping tools	N	N	L	M	H	VH
	Availability and utilization of usability labs	N	N	N	N	H	VH
Methods	Utilization of appropriate UCD methods	N	N	L	M	H	VH
Management UCD Awareness	Awareness of the importance of the UCD practitioner role	N	N	L	M	H	VH
	Management understand that user centered design must be part of the business strategy	N	N	L	M	H	VH
	Management support the inclusion and prioritization of UCD activities in the agile development process	N	N	L	M	H	VH
Training	UCD Awareness Training to Developers	N	N	N	N	H	VH
	UCD Awareness Training to Agile coaches/scrum master and product owner	N	N	N	N	H	VH
	UCD Awareness Training to Customers	N	N	N	N	H	VH
	UCD Awareness Training to Business Managers	N	N	N	N	H	VH
	UCD Practitioner Role Awareness Training to Developers	N	N	N	N	H	VH
	UCD Practitioner Role Awareness Training to Agile coaches/scrum master and product owner	N	N	N	N	H	VH
	UCD Practitioner Role Awareness Training to Customers	N	N	N	N	H	VH
	UCD Practitioner Role Awareness Training to Business Managers	N	N	N	N	H	VH
	Agile Awareness Training to UCD Practitioners	N	N	N	N	H	VH
	Agile Awareness Training to Customers	N	N	N	N	H	VH
	Agile Awareness Training to Business Managers	N	N	N	N	H	VH
	Agile Developer Role Awareness Training to UCD Practitioners	N	N	N	N	H	VH
Standards, Patterns, Style Guides	Utilization of UCD standards	N	N	N	M	H	VH
	Utilization of user interface patterns	N	N	N	M	H	VH
	Utilization of UCD style guides	N	N	N	M	H	VH
Workspace	Colocation of developers and UCD practitioners	N	N	N	N	H	VH

Table K.2: Agile User Centred Design Integration Maturity Levels Ratings for UCD Infrastructure Dimension

Table K.3 reflects the AUCDI maturity model ratings sheet for Cycle N-1 of the AUCDI process as the second dimension of the AUCDI maturity model.

Process	Practice	ML0	ML1	ML2	ML3	ML4	ML5
Plan the UCD Process	Identify and plan customer involvement	N	N	L	M	H	VH
	Identify and plan user involvement	N	N	L	M	H	VH
	Select user centered design methods	N	N	L	M	H	VH
	Identify the relevant UCD specialist skills required and plan to provide them	N	N	L	M	H	VH
User Analysis-User Groups	Identification and understanding of user groups	N	N	L	M	H	VH
User Analysis-Context Of Use	Identification and understanding of user characteristics	N	N	L	M	H	VH
	Identification and understanding of user tasks	N	N	L	M	H	VH
	Identification and understanding of user goals	N	N	L	M	H	VH
	Identification and understanding of the technical environment of old product	N	N	L	M	H	VH
	Identification and understanding of the organizational environment of old product	N	N	L	M	H	VH
	Identification and understanding of the physical environment of old product	N	N	L	M	H	VH
	Identification and understanding of the technical environment of new product	N	N	L	M	H	VH
	Identification and understanding of the organizational environment of new product	N	N	L	M	H	VH
	Identification and understanding of the physical environment of new product	N	N	L	M	H	VH
Task Analysis	Identification and understanding of the non-functional tasks' attributes	N	N	L	M	H	VH
	Identification and understanding of problems that users meet when performing tasks	N	N	L	M	H	VH
	Identification and understanding of risks that users meet when performing a task	N	N	L	M	H	VH
User Requirements	Identification and understanding of usability and/or user experience requirements	N	N	L	M	H	VH
UI Design Requirements	Identification and understanding of UI design requirements	N	N	L	M	H	VH
Lightweight Documentation	Maintaining lightweight documentation for results of UCD process planning, user analysis and task analysis, user and UI design requirements	N	N	L	M	H	VH

Table K.3: Agile User Centred Design Integration Maturity Levels Ratings Sheet for AUCDI Process Dimension - Cycle N-1

Table K.4 reflects the AUCDI maturity model Rating sheet for Cycle N of the parallel track of the AUCDI process.

Process	Practice	ML0	ML1	ML2	ML3	ML4	ML5
Synchronization Efforts	Developers share product vision with UCD team	N	N	N	M	H	VH
	UCD practitioner share user information with developers	N	N	N	M	H	VH
	UCD practitioner share task analysis information with developers	N	N	N	M	H	VH
	UCD practitioner share UCD vision with developers	N	N	N	M	H	VH
—	Design Chunking	N	N	N	M	H	VH
—	Inclusion of UCD activities (usability/user experience features) in product backlog or user stories	N	N	L	M	H	VH
—	Prioritization of UCD activities/features in cycles (sprints/iterations)	N	N	L	M	H	VH
—	Developers implementing back-end functionality	N	N	L	M	H	VH
—	UCD practitioners performing interaction and user task design for cycle N+1	N	N	N	M	H	VH
—	UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+1	N	N	L	M	H	VH
—	UCD practitioners gathering user and UI design requirements for cycle N+2	N	N	N	M	H	VH

Table K.4: Agile User Centred Design Integration Maturity Model Rating Sheet for AUCDI Process Dimension - Parallel Track- Cycle N

Table K.5 reflects the AUCDI maturity model rating sheet for Cycle N+1 of the parallel track of the AUCDI process.

Practice	ML0	ML1	ML2	ML3	ML4	ML5
Developers implementing functionality and coding for interaction and user task designs	N	N	L	M	H	VH
Synchronization efforts - Developers understand design - UCD practitioner continually clearing up any design questions posed by developers	N	N	N	M	H	VH
UCD practitioners performing usability evaluation and analysis of usability evaluation results for cycle n code	N	N	N	M	H	VH
Synchronization efforts- UCD practitioners sharing usability evaluation results for cycle n code with the development team	N	N	N	M	H	VH
UCD practitioners performing interaction and user task design for cycle N+2	N	N	N	M	H	VH
UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+2	N	N	L	M	H	VH
UCD practitioners gathering user and UI design requirements for cycle N+3	N	N	N	M	H	VH

Table K.5: Agile User Centred Design Integration Maturity Model Rating Sheet for AUCDI Process Dimension - Parallel Track- Cycle N+1

Table K.6 reflects the AUCDI maturity model rating sheet for Cycle N+2 of the parallel track of the AUCDI process.

Practice	ML0	ML1	ML2	ML3	ML4	ML5
Developers implementing functionality and coding for interaction and user task designs	N	N	L	M	H	VH
Synchronization efforts - Developers understand design - UCD practitioner continually clearing up any design questions posed by developers	N	N	N	M	H	VH
UCD practitioners performing usability evaluation and analysis of usability evaluation results for cycle N+1 code	N	N	N	M	H	VH
Synchronization efforts - UCD practitioners sharing usability evaluation results for cycle N code with the development team	N	N	N	M	H	VH
UCD practitioners performing interaction and user task design for cycle N+3	N	N	N	M	H	VH
UCD practitioners iteratively redoing interaction and user task design for cycle n code according to usability evaluation results acquired in cycle N+1	N	N	N	M	H	VH
UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+3	N	N	L	M	H	VH
UCD practitioners gathering user and UI design requirements for cycle N+4	N	N	N	M	H	VH

Table K.6: Agile User Centred Design Integration Maturity Model Rating Sheet for AUCDI Process Dimension - Parallel Track- Cycle N+2

Table K.7 reflects the AUCDI maturity model rating sheet for the people dimension as the third dimension of the AUCDI maturity model.

Person	Practices	ML0	ML1	ML2	ML3	ML4	ML5
Customers	Identification and selection of customers	N	N	L	M	H	VH
	Awareness of Agile	N	N	L	M	H	VH
	Awareness of User Centered Design	N	N	L	M	H	VH
	Early Involvement	N	N	L	M	H	VH
	Continual Involvement	N	N	L	M	H	VH
Users	Identification and selection of users	N	N	L	M	H	VH
	Early Involvement	N	N	L	M	H	VH
	Continual Involvement	N	N	L	M	H	VH
UCD Practitioner	Competence	N	N	N	M	H	VH
	Awareness- Understanding of Agile values, principles and practices	N	N	N	M	H	VH
	Awareness- Understanding of Agile developer role	N	N	N	M	H	VH
	Attitude- Continuous communication	N	N	N	M	H	VH
	Attitude- Respect and trust for developers	N	N	N	M	H	VH
	Attitude- Maintaining visibility of UCD team work throughout the agile development process	N	N	N	M	H	VH
Developers	Communication skills	N	VL	L	M	H	VH
	Awareness- Understanding of UCD	N	VL	L	M	H	VH
	Awareness- Understanding of UCD practitioner role	N	VL	L	M	H	VH
	Attitude- Respect and trust for UCD practitioner	N	N	L	M	H	VH
	Attitude- Trust customer	N	N	L	M	H	VH
Coach or Scrum Master and Prod.Owner	Communication skills	N	N	L	M	H	VH
	Awareness- Understanding of UCD	N	N	L	M	H	VH
	Awareness- Understanding of UCD practitioner role	N	N	L	M	H	VH
	Attitude- Respect and trust for UCD practitioner	N	N	N	M	H	VH
	Attitude- Trust customer	N	N	L	M	H	VH

Table K.7: Agile User Centred Design Integration Maturity Model Rating Sheet for People Dimension

Table K.8 reflects the AUCDI maturity model rating sheet for the UCD continuous improvement as the fourth dimension of the AUCDI maturity model.

Practice	**ML0**	**ML1**	**ML2**	**ML3**	**ML4**	**ML5**
Presence of a UCD monitoring process across projects	N	N	N	N	N	VH
Presence of a UCD systematic improvement process for UCD activities, tools, methods, workspace, skills and awareness	N	N	N	N	N	VH
Benchmarking product's usability and/or user experience against competitive products' usability and/or user experience	N	N	N	N	N	VH
Product decisions emerge from end user and customer studies and are targeted to meet users' needs and expectations	N	N	N	N	N	VH

Table K.8: Agile User Centred Design Integration Maturity Model Rating Sheet for UCD Continuous Improvement Dimension

K.11 AUCDI Maturity Model Recording sheet

Table K.9 reflects the AUCDI maturity model recording sheet for the UCD infrastructure as the first dimension of the AUCDI maturity model.

Item	Practices	Rating
Funds	Presence/utilization of dedicated funding for early user focus, field studies, usability and / or user experience activities, etc.	
Staff	Presence of qualified UCD team led by UCD manager Presence of qualified UCD practitioner per project/in this project	
Tools	Availability and utilization of prototyping tools Availability and utilization of usability labs	
Methods	Utilization of appropriate UCD methods	
Management UCD Awareness	Awareness of the importance of the UCD practitioner role Management understand that user centered design must be part of the business strategy Management support the inclusion and prioritization of UCD activities in the agile development process	
Training	UCD Awareness Training to Developers UCD Awareness Training to Agile coaches/scrum master and product owner UCD Awareness Training to Customers UCD Awareness Training to Business Managers UCD Practitioner Role Awareness Training to Developers UCD Practitioner Role Awareness Training to Agile coaches/scrum master and product owner UCD Practitioner Role Awareness Training to Customers UCD Practitioner Role Awareness Training to Business Managers Agile Awareness Training to UCD Practitioners Agile Awareness Training to Customers Agile Awareness Training to Business Managers Agile Developer Role Awareness Training to UCD Practitioners	
Standards, Patterns, Style Guides	Utilization of UCD standards Utilization of user interface patterns Utilization of UCD style guides	
Workspace	Colocation of developers and UCD practitioners	

Table K.9: Agile User Centred Design Integration Maturity Model Recording Sheet for UCD Infrastructure Dimension

Table K.10 reflects the AUCDI maturity model recording sheet for Cycle N-1 of the AUCDI process as the second dimension of the AUCDI maturity model.

Process	**Practice**	**Rating**
Plan the UCD Process	Identify and plan customer involvement Identify and plan user involvement Select user centered design methods Identify the relevant UCD specialist skills required and plan to provide them	
User Groups	Identification and understanding of user groups	
Context Of Use	Identification and understanding of user characteristics Identification and understanding of user tasks Identification and understanding of user goals Identification and understanding of the technical environment of old product Identification and understanding of the organizational environment of old product Identification and understanding of the physical environment of old product Identification and understanding of the technical environment of new product Identification and understanding of the organizational environment of new product Identification and understanding of the physical environment of new product	
Task Analysis	Identification and understanding of the non-functional tasks' attributes Identification and understanding of problems that users meet when performing tasks Identification and understanding of risks that users meet when performing a task	
User Requirements	Identification and understanding of usability and/or user experience requirements	
UI Design Requirements	Identification and understanding of UI design requirements	
Lightweight Documentation	Maintaining lightweight documentation for results of UCD process planning, user analysis and task analysis, user and UI design requirements	

Table K.10: Agile User Centred Design Integration Maturity Model Recording Sheet for AUCDI Process Dimension - Cycle N-1

Table K.11 reflects the AUCDI maturity model recording sheet for Cycle N of the parallel track of the AUCDI process.

Process	**Practice**	**Rating**
Synchronization Efforts	Developers share product vision with UCD team UCD practitioner share user information with developers UCD practitioner share task analysis information with developers UCD practitioner share UCD vision with developers	
—	Design Chunking	
—	Inclusion of UCD activities (usability/user experience features) in product backlog or user stories	
—	Prioritization of UCD activities/features in cycles (sprints/iterations)	
—	Developers implementing back-end functionality	
—	UCD practitioners performing interaction and user task design for cycle N+1	
—	UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+1	
—	UCD practitioners gathering user and UI design requirements for cycle N+2	

Table K.11: Agile User Centred Design Integration Maturity Model Recording Sheet for AUCDI Process Dimension - Parallel Track- Cycle N

Table K.12 reflects the AUCDI maturity model recording sheet for Cycle N+1 of the parallel track of the AUCDI process.

Process	Practice	Rating
—	Developers implementing functionality and coding for interaction and user task designs	
—	Synchronization efforts - Developers understand design - UCD practitioner continually clearing up any design questions posed by developers	
—	UCD practitioners performing usability evaluation and analysis of usability evaluation results for cycle n code	
—	Synchronization efforts- UCD practitioners sharing usability evaluation results for cycle n code with the development team	
—	UCD practitioners performing interaction and user task design for cycle N+2	
—	UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+2	
—	UCD practitioners gathering user and UI design requirements for cycle N+3	

Table K.12: Agile User Centred Design Integration Maturity Model Recording Sheet for AUCDI Process Dimension - Parallel Track- Cycle N+1

Table K.13 reflects the AUCDI maturity model recording sheet for Cycle N+2 of the parallel track of the AUCDI process.

Process	Practice	Rating
—	Developers implementing functionality and coding for interaction and user task designs	
—	Synchronization efforts - Developers understand design - UCD practitioner continually clearing up any design questions posed by developers	
—	UCD practitioners performing usability evaluation and analysis of usability evaluation results for cycle N+1 code	
—	Synchronization efforts - UCD practitioners sharing usability evaluation results for cycle N code with the development team	
—	UCD practitioners performing interaction and user task design for cycle N+3	
—	UCD practitioners iteratively redoing interaction and user task design for cycle n code according to usability evaluation results acquired in cycle N+1	
—	UCD practitioners maintaining lightweight documentation of design and design rationale for cycle N+3	
—	UCD practitioners gathering user and UI design requirements for cycle N+4	

Table K.13: Agile User Centred Design Integration Maturity Model Recording Sheet for AUCDI Process Dimension - Parallel Track- Cycle N+2

Table K.14 reflects the AUCDI maturity model recording sheet for the people dimension as the third dimension of the AUCDI maturity model.

Person	Practices	Rating
Customers	Identification and selection of customers Awareness of Agile Awareness of User Centered Design Early Involvement Continual Involvement	
Users	Identification and selection of users Early Involvement Continual Involvement	
UCD Practitioner	Competence Awareness- Understanding of Agile values, principles and practices Awareness- Understanding of Agile developer role Attitude- Continuous communication Attitude- Respect and trust for developers Attitude- Maintaining visibility of UCD team work throughout the agile development process	
Developers	Communication skills Awareness- Understanding of UCD Awareness- Understanding of UCD practitioner role Attitude- Respect and trust for UCD practitioner Attitude- Trust customer	
Coach/Scrum Master and Product Owner	Communication skills Awareness- Understanding of UCD Awareness- Understanding of UCD practitioner role Attitude- Respect and trust for UCD practitioner Attitude- Trust customer	

Table K.14: Agile User Centred Design Integration Maturity Model Recording Sheet for People Dimension

Table K.15 reflects the AUCDI maturity model recording sheet for the UCD continuous improvement as the fourth dimension of the AUCDI maturity model.

Practice	**Rating**
Presence of a UCD monitoring process across projects	
Presence of a UCD systematic improvement process for UCD activities, tools, methods, workspace, skills and awareness	
Benchmarking product's usability and/or user experience against competitive products' usability and/or user experience	
Product decisions emerge from end user and customer studies and are targeted to meet users' needs and expectations	

Table K.15: Agile User Centred Design Integration Maturity Model Recording Sheet for UCD Continuous Improvement Dimension

K.12 Assessment Indicators: Typical Quotes

This appendix provides an example for a typical quote that may be used by the interviewed members of the assessed organisations. The AUCDIMM assessor should use these quotes in order to ensure that he has correctly assessed the organisational AUCDI maturity level. The inclusion of typical quotes as a guideline for assessors was used in Crosby [49], Earthy [68] in Quality Maturity Grid and UMM-HCS respectively.

Level 0: Not Possible

"We do not have a problem with user centred design/usability."

Level 1: Possible

"I would like to integrate UCD and Agile development processes but I do not know how."

"I am trying to focus more on users in this project." - Statement by the developers since there is no UCD practitioners available.

Level 2: Encouraged

"We should focus more on user centred design/usability." - Statement by management staff.

Level 3: Enabled/Practiced

"We should allow more time for frequent usability testing."

"We should have a user pool."

"We should sign a contract with a user recruitment company."

"Are projects making use of the usability lab?"

Level 4: Managed

"Assign a mentor for the new UCD practitioner."

"Schedule for the periodic UCD training for developers, Agile coach or scrum master and product manager."

Level 5: Continuous Improvement

"Decrease the workload on the UCD practitioners - Assign them to less projects."

"We need to acquire a more spacious room to collocate developers and UCD practitioners."

K.13 Assessment Guidelines

A set of guidelines for performing the assessment were put together in order to provide assessors by a clear road map for conducting the assessment. Those assessment guidelines are as follows:

- Before initiating the assessment the assessor should ensure the confidentiality of the results to the stakeholders of the assessed organisation and choose a benchmark project that has attempted AUCDI and utilise the model in assessing it.
- The assessment should occur via interviews with selected staff. This staff is presented by UCD team manager, UCD practitioner, one or two developers, Agile coach in case of XP and Scrum master and product owner in case of Scrum, one management staff, and customer if possible.
- The assessor should record the answers in the AUCDI maturity model recording sheet included in section K.11.
- After the assessment of the benchmark project, other projects should be chosen to be assessed, their number is dependent on the organisation size and its cooperation in dedicating time for the assessment and staff inclusion in the assessment.
- Each interview duration should not exceed an hour.
- The rating system to be used is: very high (>4.5), high (3.5-4.5), medium (2.5-3.5) and low (1.5-2.5), very low (>0 and <1.5), No (0).
- The interview is preferably conducted on site.
- The interview is started by the assessor introducing the model and its benefits and aims then by mentioning the agreed upon benchmark project and declaring that all questions posed will be focused on that particular project.
- The assessment procedure should be based upon a review of any material gathered by the organisation in support of its claim in regards to the key dimensions and practices assessed.
- During the interview the assessor should try to match interviewees' answers to the typical maturity level sample quotes included in section K.12.
- The assessor should rate each key dimension and its included practices via asking questions and asking for evidence.
- In case of the lack of occurrence or absence of a particular practice the assessor should record "No" (0) in the scoring sheet.

- In case of the interviewee's inability to answer one of the questions the assessor should record "don't know" in the scoring sheet.
- The AUCDI maturity levels with their corresponding performance rating included in section K.10 should be used by the assessor for comparing the recorded scoring from the maturity recording sheet with the AUCDI maturity levels in order to determine their organisational AUCDI maturity level.
- After finishing the assessment of the benchmark project the assessor should move to the assessment of the following project.
- The assessment results should be used by the assessor and the assessed organisation to review or improve the AUCDI within the organisation.

K.14 Request for AUCDI Expert Evaluation

Subject : Acting as an Expert Reviewer for a maturity model for Agile and User Centred Design Integration (AUCDI)

My name is Dina Salah, I am a PhD student, at the university of York, working in the area of integrating Agile development processes and user centred design. I am supervised by Professor Richard Paige and Dr. Paul Cairns. I am contacting you as an expert in the domain of integrating Agile development processes and user-centred design. My PhD has resulted in developing a multidimensional maturity model for integrating Agile development processes and user centred design that can can help organisations assess their current status and the factors that impact AUCDI process and pinpoint weaknesses and strengths in order to plan for improvement.

The first phase of evaluating this AUCDI maturity model is via a domain expert panel. You have been selected as one of the experts in the domain of integrating Agile and user centred design and I would be most delighted if you would agree to be one of the members of the expert panel that will evaluate this model. The evaluation process involves sending the model to you via email in order to be examined and then you can kindly fill a two page evaluation survey that evaluates the model's usefulness, ease of use, comprehensiveness, practicality, etc.

The estimated time that is needed for the expert evaluation is 2-3 hours (including a short briefing that I give about the model, examining the model and filling the survey).

Your feedback on the model will contribute to its evolution into an improved version and this will be acknowledged in my thesis and in any published papers that reports on the model evaluation (unless you prefer your information to remain anonymous).

I would be glad to provide you with any further details.

Best Regards,

Dina Salah

K.15 Informed Consent Form for Domain Expert Evaluation for AUCDI Maturity Model

University of York Informed Consent Form for Domain Expert Evaluation for Agile and User Centred Design Integration Maturity Model

Title of Project: A Maturity Model for Integrating Agile Development Processes and User Centred Design

Principal Investigator: Dina Salah

We invite you to take part in a research study at the University of York. This study is focused on maturity models in the domain of integrating Agile development processes and UCD.

1. **Purpose of the Study**

   The purpose of this study is to evaluate a maturity model for integrating Agile development processes and UCD that I have developed as a result of my PhD. The PhD has resulted in developing a method independent process model that cover AUCDI related activities throughout the Agile development life cycle. This AUCDI process model includes detailed activities, work products, work flows, roles, and responsibilities. The AUCDI process model was used as the basis for implementing a multidimensional AUCDI maturity model that is composed of three components: first, a multidimensional reference model that constitutes a set of fundamental elements that affect AUCDI and thus should be examined in an assessment. Second, a performance scale to rate the project's and organisation's performance in the assessed elements included in the AUCDI reference model. Third, assessment guidelines that provide practical guidance for performing the assessment. The results of the assessment can help organisations assess their current status and the factors that impact AUCDI process and pinpoint weaknesses and strengths in order to plan for improvement.

2. **Procedures**

   The first phase of evaluating this AUCDI maturity model is via a domain expert panel. You have been selected as one of the experts in the domain of integrating Agile and UCD. The evaluation process involves examining the model and filling a two page evaluation survey that evaluates the model's usefulness, ease of use, comprehensiveness, practicality, etc.

3. **Time Duration of the Procedures**

   If you agree to take part in this study, the approximate time for completion of the model's review as well as the model's evaluation survey is expected to be two hours.

4. **Potential Benefits and Risks**

   Your participation in the model's evaluation will be acknowledged in my thesis and in any published papers that reports on the model evaluation (unless you prefer your information to remain anonymous). There are no known risks to participation in this study.

5. **Statement of Confidentiality**

   If you prefer anonymity then all data that identifies you that is gathered from the evaluation will be treated in a confidential manner: It will be archived in a secure location and will only be accessed by myself and my supervisory team for the purposes of the analysis. When your data are reported or described, all identifying information will be removed. You may also request for destroying the evaluation forms if you wish so at a later date.

6. **Voluntary Participation**

   Your participation in this study is entirely voluntary and you are free to withdraw at any point and you do not need to give a reason for withdrawal.

7. **Contact Information for Questions or Concerns**

   You have the right to ask any question you may have in regards to this research. If you have any questions, complaints or concerns, kindly contact Dina Salah, dm560@york.ac.uk. If you would like to contact one of my supervisors you can contact professor Richard Paige, richard.paige@york.ac.uk or Dr. Paul Cairns, paul.cairns@york.ac.uk

8. **Signature and Consent to Participate in the Study**

   By signing this consent form, you indicate that you have understood what it means to take part in this study and that you are voluntarily choosing to take part in this research study and participate as a member of the expert panel for evaluating the Agile and UCD integration maturity model.

Signature of Expert Reviewer

Date

K.16 AUCDI Domain Expert Evaluation Survey

Figure K.3 shows the AUCDI domain expert evaluation form that was designed in order to be used to conduct the evaluation

THE UNIVERSITY of York

Computer Science Department

Agile and User Centered Design integration Maturity Model (AUCDIMM) – AUCDI Domain Expert Evaluation Forms

Expert Information					
Date					
Name (Optional)					
Organization/Institute					
Position					
Email					
Criteria	**Strongly Disagree**	**Slightly Disagree**	**Neither Disagree Nor Agree**	**Slightly Agree**	**Strongly Agree**
Maturity Levels					
The maturity levels are sufficient to represent, all maturation stages of AUCDI (Sufficiency)					
There is no overlap detected between descriptions of maturity levels (Accuracy)					
Processes and Practices					
The processes and practices are relevant to AUCDI (Relevance)					
Processes and practices cover all aspects impacting/ involved in AUCDI (Comprehensiveness)					
Processes and practices are clearly distinct (Mutual Exclusion)					
Processes and practices are correctly assigned to their respective maturity level (Accuracy)					
AUCDI Maturity Model					
Understandability					
The maturity levels are understandable					
The assessment guidelines are understandable					
The documentation is understandable					
Ease of Use					
The scoring scheme is easy to use					
The assessment guidelines are easy to use					
The documentation is easy to use					
Usefulness					
The maturity model is useful for providing organisations with a set of dimensions, processes, and practices that act as a road map for successful AUCDI.					
The maturity model is useful for providing organisations with an understanding of AUCDI roles, timing, responsibilities, success factors, and challenges					
The maturity model is useful for providing organisations with a diagnostic tool to assess AUCDI challenges that could develop during the AUCDI process					
The maturity model is useful for providing organisations with a diagnostic tool to assess AUCDI success factors					

Q1. Would you add any maturity levels? If so please explain what and why?
Q2. Would you update the maturity level description? If so please explain what and why?
Q3. Would you add any processes or practices? If so please explain what and why?
Q4. Would you remove any of the processes or practices? If so please explain what and why?
Q5. Would you redefine/update any of the processes or practices? If so please explain what and why?
Q6. Would you suggest any updates or improvements related to the scoring scheme? If so please explain what and why?
Q7. Would you suggest any updates or improvement related to the assessment guidelines? If so please explain what and why?
Q8. Would you like to elaborate on any of your answers?
Q9. Could the model be made more useful? How?
Q10. Could the model be made more practical? How?
Q11. Are you interested in further usage of the model in a case study evaluation?

Figure K.3: Agile and User Centred Design Integration Domain Expert Evaluation Forms

K.17 Evolution of Agile and User Centred Design Integration Maturity Model

AUCDI maturity model evaluation involves three phase process: author evaluation, AUCDI domain expert reviews and case study evaluation. The results of this evaluation led to the evolution of the model into a number of subsequent versions as shown in table K.16. This section shows the changes to the model's maturity levels, key practices, scoring scheme, and assessment guidelines and the author, expert reviewer or case study details and reasons behind these changes.

Version	Date	Reviewer
1.0	7/3/2013	Dr. Paul Cairns
2.0	21/4/2014	Mona Singh
2.1	23/4/2014	Jim Ungar
2.2	30/4/2014	Jason Chong Lee

Table K.16: Evolution of Agile User Centred Design Integration Maturity Model

The valuable evaluation that was received from expert reviewers were used to refine the model as follows:

Version 1.0:

Based on feedback from PhD Supervisor, Dr. Paul Cairns, the questions in AUCDI domain expert evaluation forms were transfered into statements, using simpler terms in question wording, and adding questions to elaborate on reviewer's answers.

Version 2.0 to 2.2

Based on feedback from the AUCDI domain experts the following changes were made:

- Maturity level 5 description was updated to include a process for reviewing and assessing guidelines.
- Training practices were updated to include UCD awareness training to product owner. Also further product owner features were added to the people dimension. Those included understanding of UCD and understanding of UCD practitioner role.
- The description of standards, patterns, and style guides was edited to indicate their role in ensuring consistency across products and re-usability

as well as their importance in improving projects developed by small agile teams.

- Early work phase description was edited to refer to the communication between UCD practitioners and architects to check the project's technical feasibility. Moreover, the description was edited to include its utilisation in acquiring feedback from management and sales department.

References

[1] Abrahamsson, P., A. Hanhineva, H. Hulkko, T. Ihme, J. Jaalinoja, M. Korkala, J. Koskela, P. Kyllonen, and O. Salo (2004). Mobile-D: An Agile Approach for Mobile Application Development. In *Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications*, OOPSLA '04, New York, NY, USA, pp. 174–175. ACM. 88

[2] Abrahamsson, P., O. Salo, J. Ronkainen, and J. Warsta (2002). Agile Software Development Methods: Review and Analysis. Technical report, VTT Finland. 34

[3] Adikari, S., C. McDonald, and J. Campbell (2009). Little Design Up-Front: A Design Science Approach to Integrating Usability into Agile Requirements Engineering. In J. Jacko (Ed.), *Human-Computer Interaction. New Trends*, Volume 5610 of *Lecture Notes in Computer Science*, pp. 549–558. Springer Berlin / Heidelberg. 54, 87, 93, 94, 98, 101, 102, 103, 112, 122, 153, 368

[4] Aigner, W. (2009). Combining Extreme Programming and Interaction Design. 88

[5] Albisetti, M. (2010). Launchpad's Quest for a Better and Agile User Interface. In *11th International Conference on Agile Software Developement, XP2010*, Trondheim, Norway. 54, 93, 94, 98, 100, 101, 102, 110, 111, 116, 118, 122, 125, 126, 129, 132, 164, 170, 179, 318, 370

[6] Ambler, S. (2006a). Agile Modeling. http://www.agilemodeling.com. 32

[7] Ambler, S. (2006b). Introduction to Agile Usability-User Experience (UX) Activities on Agile Development Projects. 88

[8] Ambler, S. (2008). Tailoring Usability into Agile Software Development Projects. In E. Law, E. Hvannberg, and G. Cockton (Eds.), *Maturing Usability*, Human Computer Interaction Series, pp. 75–95. Springer. London, UK. 54, 91, 93, 95, 98, 100, 101, 103, 115, 117, 120, 307

[9] Armitage, J. (2004, January). Are Agile Methods Good for Design? *Interactions 11*(1), 14–23. 19, 54, 93, 95, 98, 100, 101, 102, 109, 110, 112, 370

[10] Avison, D. and G. Fitzgerald (2003, January). Where Now for Development Methodologies? *Communications of the ACM 46*(1), 78–82. 29, 406

[11] Avison, D., F. Lau, M. Myers, and P. A. Nielsen (1999, January). Action Research. *Communications of ACM 42*(1), 94–97. 135

[12] Barnett, L. (2007, August). Agile Survey Results:Widespread Adoption. Agile Journal and VersionOne. 29

[13] Bass, L. and R. Kazman (2003). Bridging the Process and Practice Gaps between Software Engineering and Human Computer Interaction. *Software Process Improvement and Practice Journal 8(2)*, pp.63–131. 53

[14] Beck, K. (2000). Manifesto for Agile Software Developement. 18, 30, 31, 106, 114, 130, 176, 177, 311, 416, 419, 432

[15] Beck, K. (2004). *Extreme Programming Explained. Embrace Change* (2nd Edition ed.). Boston, MA: Addison Wesley Longman Publishing Co., Inc. 32, 110, 130, 313

[16] Becker, J., R. Knackstedt, and J. Poppelbus (2009). Developing Maturity Models for IT Management. *Business and Information Systems Engineering 1*, 213–222. 56, 58, 192, 221, 402, 413, 416, 419

[17] Becker, J., B. Niehaves, J. Poppelbus, and A. Simon (2010). Maturity Models in IS Research. In *ECIS 2010*, Pretoria. 9, 57, 58

[18] Benbasat, I., A. Dexter, D. Drury, and R. Goldstein (1984, May). A Critque of the Stage Hypothesis: Theory and Empirical Evidence. *Communications of the ACM 27*(5), 476–485. 224

[19] Bevan, N. and A. M (1997). Quality In Use:Incorporating Human Factors into the Software Engineering Life Cycle. In *3rd IEEE International Software Engineering Standards Symposium and Forum*. 35

[20] Beyer, H., K. Holtzblatt, and L. Baker (2004). An Agile Customer-Centered Method: Rapid Contextual Design. In *XP/AU*. 19, 44, 54, 55, 89, 91, 93, 94, 98, 100, 101, 103, 111, 222, 417

[21] Beyer, H., K. Holtzblatt, and L. Baker (2008). An Agile User-Centered Method: Rapid Contextual Design. 44, 89

[22] Biolchini, J., P. G. Mian, A. C. C. Natali, and G. H. Travassos (2005). Systematic Review in Software Engineering. Technical report, COPPEUFRJ , Systems Engineering and computer Science Department, Rio De Janeiro. 67, 72, 74, 78

[23] Blomkvist, S. (2005). Towards a Model for Bridging Agile Development and User-Centered Design. In A. Seffah, J. Gulliksen, and M. Desmarais (Eds.),

Human-Centered Software Engineering Ů Integrating Usability in the Software Development Lifecycle, Volume 8 of *HumanŰComputer Interaction Series*, pp. 219–244. Springer Netherlands. 18, 29, 30, 38, 39, 54, 93, 95, 98, 100, 101, 103, 106, 109, 124, 126, 314, 316, 368, 402, 408, 414

[24] Boehm, B. and R. Turner (2004). *Balancing Agility and Discipline A Guide for the Perplexed*. Pearson Education Limited. 30, 130, 313

[25] Borup-Harning, M., R. Kazman, K. K. Daniel Kerkow, and J. Gulliksen (2005). Integrating Software Engineering and Usability Engineering. In *INTERACT 2005*. 52

[26] Borup-Harning, M. and J. Vanderdonckt (2003). Closing the Gaps: Software Engineering and Human-Computer Interaction. In *INTERACT*. 52

[27] Bosch, J. and P. Bosch-Sijtsema (2011, July). Introducing Agile Customer-Centered Development in a Legacy Software Product Line. *Software Practice and Experience 41*(8), 871–882. 88

[28] Bourimi, M., T. Barth, J. Haake, B. Ueberschär, and D. Kesdogan (2010). AFFINE for Enforcing Earlier Consideration of NFRS and Human Factors when Building Socio-Technical Systems Following Agile Methodologies. In *Proceedings of the Third International Conference on Human-Centred Software Engineering*, HCSE'10, Berlin, Heidelberg, pp. 182–189. Springer-Verlag. 91, 368

[29] Boyatzis, R. (1998). *Transforming Qualitative Information: Thematic Analysis and Code Development*. Thousand Oaks, CA: SAGE. 142

[30] Braun, V. and V. Clarke (2006). Using Thematic Analysis in Psychology. *Qualitative Research in Psychology 3 (2)*(ISSN 1478-0887), pp. 77–101. 79, 142

[31] Brereton, P., B. Kitchenham, D. Budgen, M. Turner, and M. Khalil (2007, April). Lessons from Applying the Systematic Literature Review Process within the Software Engineering Domain. *Journal of Systems and Software 80*(4), 571–583. 74, 79

[32] Broschinsky, D. and L. Baker (2008). Using Persona with XP at LANDesk Software, an Avocent Company. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 543–548. IEEE Computer Society. 54, 93, 94, 98, 100, 101, 102, 104, 111, 112, 118, 128, 131, 132, 161, 370

[33] Brown, J., G. Lindgaard, and R. Biddle (2008). Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artefacts. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 39–50. IEEE Computer Society. 93, 94, 98, 99, 100, 101, 102, 116, 131, 368

[34] Brown, J., G. Lindgaard, and R. Biddle (2011, aug.). Collaborative Events and Shared Artefacts: Agile Interaction Designers and Developers Working

Toward Common Aims. In *Agile Conference (AGILE), 2011*, pp. 87 –96. 54, 93, 94, 98, 100, 101, 102, 114, 116, 118, 124, 306, 316, 368, 428

[35] Budwig, M., S. Jeong, and K. Kelkar (2009). When User Experience Met Agile: A Case Study. In *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, New York, NY, USA, pp. 3075–3084. ACM. 93, 94, 98, 100, 101, 102, 104, 107, 111, 113, 118, 128, 129, 131, 132, 302, 306, 307, 308, 370

[36] Carbon, R., J. Dorr, and M. Trapp (2004). Focusing Extreme Programming on Usability. In *GI Jahrestagung*, pp. 147–152. 54, 55, 93, 94, 98, 100, 101, 102, 103, 122, 170, 368

[37] Carroll, J. (2000a). Introduction to the Special Issue on Scenario Based Developement. *Interacting with Computers 13(1)*, 41–42. 45, 405

[38] Carroll, J. (2000b). *Making Use: Scenario Based Design of Human Computer Interactions*. The MIT Press.Cambridge,MA. 45, 405

[39] Chamberlain, S., H. Sharp, and N. Maiden (2006). Towards a Framework for Integrating agile Development and User-Centred Design. In *Proceedings of the 7th International Conference on Extreme Programming and Agile Processes in Software Engineering*, XP'06, Berlin, Heidelberg, pp. 143–153. Springer-Verlag. 18, 38, 54, 93, 94, 98, 99, 100, 101, 102, 106, 107, 109, 110, 111, 121, 302, 368, 408, 415

[40] Coallier, F. (1995). TRILLIUM: A Model for the Assessment of Telecom Product Development & Support Capability. Software Process Newsletter 2.pp.13-8. 186

[41] Coatta, T. and R. Rutter (2011). UX Design and Agile: A Natural Fit? *Communications of the ACM 54*(1), 54–60. 93, 95, 98, 100, 101, 102, 111, 112, 114, 117, 121, 127, 318, 370

[42] Cockburn, A. (2004). *Crystal Clear : A Human-Powered Methodology for Small Teams*. Addison-Wesley Professional. 32, 55, 56

[43] Coleman, G. and R. O'Connor (2007). Using Grounded Theory to Understand Software Process improvement: A Study of Irish Software Product Companies. *Information and Software Technology 49 (6)*, pp. 654–667. 143

[44] Constantine, L. (2001, April). Agile Usage-Centered Design. 88

[45] Constantine, L. and L. Lockwood (2002a). Process Agility and Software Usability: Towards Lightweight Usage Centred Design. Information Age 8.8. 19, 54, 55, 93, 95, 98, 99, 100, 101, 103, 112, 222, 370

[46] Constantine, L. and L. Lockwood (2002b, March). Usage-Centered Engineering for Web Applications. *IEEE Software 19*(2), 42–50. 109

[47] Constantine, L. and L. Lockwood (2003, may). Usage-Centered Software Engineering: An Agile Approach to Integrating Users, User Interfaces, and Usability into Software Engineering Practice. In *International Conference on Software Engineering*, pp. 746 –747. 54, 88

[48] Cooper, A. (2008). The Wisdom of Experience.Keynote Talk at Agile 2008 Conference. Toronto. Canada. 106

[49] Crosby, P. (1978). *Quality is Free: The Art of Making Quality Certain*. New York. USA: McGraw Hill. 185, 231, 283, 349, 438, 454

[50] Cruzes, D. and T. Dyba (2011). Recommended Steps for Thematic Synthesis in Software Engineering. In *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement*, ESEM '11, Washington, DC, USA, pp. 275–284. IEEE Computer Society. 9, 79, 80

[51] da Silva, B. S., V. C. O. Aureliano, and S. D. J. Barbosa (2006). Extreme Designing: Binding Sketching to an Interaction Model in a Streamlined HCI Design Approach. In *Proceedings of VII Brazilian Symposium on Human Factors in Computing Systems*, IHC '06, New York, NY, USA, pp. 101–109. ACM. 89, 96

[52] da Silva, T. S. (2012). *A Framework for Integrating Interaction Design and Agile Methods*. Ph. D. thesis, Pontifica Universidade Catolica Do Ro Grande Do Sul, Faculdade De Informatica. 79, 89, 96

[53] da Silva, T. S., A. Martin, F. Maurer, and M. Silveira (2011, aug.). User-Centered Design and Agile Methods: A Systematic Review. In *Agile Conference (AGILE), 2011*, pp. 77–86. 54, 67, 89, 93, 94, 96, 98, 100, 103, 368

[54] da Silva, T. S., M. Silveira, and F. Maurer (2011). Best Practices in Integrating User-Centered Design and Agile Software Development. In *Proceedings of the Companion Proceedings of the 10th Brazilian Symposium on Human Factors in Computing Systems and the 5th Latin American Conference on Human-Computer Interaction*, IHC+CLIHC '11, Porto Alegre, Brazil, Brazil, pp. 43–45. Brazilian Computer Society. 54, 88

[55] Dayton, D. and C. Barnum (2008). The Impact of Agile on UCD: Mixed Messages from a Before and After Survey. Presentation at UPA 2008. 54, 86, 88

[56] Dayton, D. and C. Barnum (2009). The Impact of Agile on User-Centered Design: Two Surveys Tell the Story. *Technical Communication 56*(3), 219–234. 86, 93, 98, 100, 101, 103, 104, 120, 121, 159, 179, 368

[57] DeBruin, T., R. Freeze, U. Kaulkarni, and M. Rosemann (2005). Understanding the Main Phases of Developing a Maturity Assessment Model. In *Australian Conference on Information Systems*, New South Wales, Sydney,Australia. 9, 58, 60, 61, 62, 63, 225, 328, 331, 332, 333, 334, 336, 337, 338, 419, 421

[58] Desilets, A. (2005, June). Are Agile and Usability Methodologies Compatible? Online. 88

[59] Detweiler, M. (2007, May). Managing UCD within Agile Projects. *Interactions 14*(3), 40–42. 18, 35, 38, 54, 93, 95, 98, 100, 101, 103, 107, 110, 111, 113, 115, 118, 119, 120, 121, 122, 123, 128, 130, 154, 159, 170, 179, 302, 306, 315, 318, 319, 369, 370, 406, 408, 414, 415, 416, 428, 434

[60] dos Santos, A. P. O. and F. Kon (2011). Applying Usability and User Experience Goals in Agile Software Development. In *XP'2011 - Proceedings of the 2nd workshop Dealing with Usability in an Agile Domain.*, Madrid, Spain. 91, 368

[61] Dubinsky, Y., S. R. Humayoun, and T. Catarci (2008). Eclipse Plug-in to Manage User Centered Design. In *I-USED 2008*. 54

[62] Duchting, M., D. Zimmermann, and K. Nebe (2007). Incorporating User Centered Requirement Engineering into Agile Software Development. In *Proceedings of the 12th international Conference on Human-Computer Interaction: Interaction Design and Usability*, HCI'07, Berlin, Heidelberg, pp. 58–67. Springer-Verlag. 54, 93, 94, 98, 100, 102, 103, 368

[63] Dyba, T. and T. Dingsoyr (2008a, August). Empirical Studies of Agile Software Development: A Systematic Review. *Information and Software Technology 50*(9-10), 833–859. 76, 79

[64] Dyba, T. and T. Dingsoyr (2008b). Strength of Evidence in Systematic Reviews in Software Engineering. In *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '08, New York, NY, USA, pp. 178–187. ACM. 66, 72, 75, 76, 79, 133, 134

[65] Dyba, T., T. Dingsoyr, and G. Hanssen (2007, sept.). Applying Systematic Reviews to Diverse Study Types: An Experience Report. In *First International Symposium on Empirical Software Engineering and Measurement, 2007. ESEM 2007.*, pp. 225 –234. 76

[66] Dyba, T., B. Kitchenham, and M. Jorgensen (2005, jan.-feb.). Evidence-Based Software Engineering for Practitioners. *IEEE Software 22*(1), 58–65. 66

[67] Earthy, J. (1996). Developement of the Usability Maturity Model: INUSE Project Deliverable D5.1.1(t). Technical report, Lloyd's Register, 71 Fenchurch St, London, EC3M 4BS. 192

[68] Earthy, J. (1998). Usability Maturity Model: Human Centredness Scale:INUSE Project Deliverable D5.1.4 (s) Version 1.2. Technical report, Llyod's Register, 71Fenchurch St, London,EC3M 4BS. 12, 13, 23, 186, 192, 231, 232, 233, 234, 235, 242, 280, 281, 284, 285, 286, 347, 349, 395, 396, 397, 398, 399, 400, 401, 438, 454

[69] Earthy, J. (2011). Usability Maturity Model: Processes Version 2.3. Technical report, Llyod's Register, 71Fenchurch St, London,EC3M 4BS. 187, 192, 402, 404, 405, 407, 413

[70] Eason and Haker (1997). User Centred Design Maturity. Internal Working Document. Technical report, Department of Human Sciences. Loughborough University. 187, 231, 283

[71] Eklund, J. and C. Levingston (2008). Usability in Agile Development. 54, 87, 88

[72] Entin, V., M. Winder, B. Zhang, and S. Christmann (2011, march). Combining Model-Based and Capture-Replay Testing Techniques of Graphical User Interfaces: An Industrial Approach. In *IEEE Fourth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pp. 572 –577. 88

[73] Ericsson, A. and H. Simon (1980). Verbal reports as data. *Psychological Review 87 (3)*, 215Ű251. 48

[74] Esteves, M. and V. Andrade (2011). Designing Interaction Concepts, Managing Customer Expectation and Mastering Agile Development in Rich Application Product Development. In *Proceedings of the 14th International Conference on Human-Computer Interaction: Design and Development Approaches - Volume Part I*, HCII'11, Berlin, Heidelberg, pp. 54–62. Springer-Verlag. 88

[75] Evnin, J. and M. Pries (2008). Are You Sure? Really? A Contextual Approach to Agile User Research. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 537–542. IEEE Computer Society. 54

[76] Federoff, M. and C. Courage (2009). Successful User Experience in an Agile Enterprise Environment. In *Proceedings of the Symposium on Human Interface 2009 on Conference Universal Access in Human-Computer Interaction. Part I: Held as Part of HCI International 2009*, Berlin, Heidelberg, pp. 233–242. Springer-Verlag. 54, 93, 94, 98, 100, 101, 102, 103, 104, 110, 111, 112, 116, 117, 120, 121, 122, 125, 126, 159, 164, 179, 306, 370

[77] Federoff, M., C. Villamor, L. Miller, J. Patton, A. Rosenstein, K. Baxter, and K. Kelkar (2008). Extreme Usability: Adapting Research Approaches for Agile Development. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, New York, NY, USA, pp. 2269–2272. ACM. 54, 88

[78] Ferreira, J. (2008). Interaction Design and Agile Development: Reconciling Iterative and Incremental Approaches. In *Agile Conference*. 54, 93, 94, 98, 100, 101, 103, 112, 120, 121, 122, 170, 308, 315, 368

[79] Ferreira, J. (2010). Work Cultures of Engagement. A Position Paper for the Workshop Dealing with Usability in an Agile Domain. In *XP*. 89, 107

[80] Ferreira, J. (2012). Agile Development and UX Design: Towards Understanding Work Cultures to Support Integration. In M. Bajec, J. Eder, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, C. Szyperski, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, and C. Szyperski (Eds.), *Advanced Information Systems Engineering Workshops*, Volume 112 of *Lecture Notes in Business Information Processing*, pp. 608–615. Springer Berlin Heidelberg. 54, 90

[81] Ferreira, J., J. Noble, and R. Biddle (2007a). Agile Development Iterations and UI Design. In *Proceedings of the AGILE 2007*, AGILE '07, Washington, DC, USA, pp. 50–58. IEEE Computer Society. 93, 94, 98, 100, 101, 102, 113, 116, 121, 154, 170, 317, 368

[82] Ferreira, J., J. Noble, and R. Biddle (2007b). Interaction Designers on eXtreme Programming Teams: Two Case Studies from the Real World. In *Proceedings of the Fifth New Zealand Computer Science Research Student Conference (NZCSRSC2007)*. 55, 93, 94, 98, 100, 101, 102, 110, 111, 113, 114, 117, 118, 124, 316, 317, 318, 319, 368, 434

[83] Ferreira, J., J. Noble, and R. Biddle (2007c). Up-front Interaction Design in Agile Development. In *Proceedings of the 8th International Conference on Agile Processes in Software Engineering and Extreme Programming*, XP'07, Berlin, Heidelberg, pp. 9–16. Springer-Verlag. 93, 94, 98, 100, 101, 102, 111, 302, 368

[84] Ferreira, J., H. Sharp, and H. Robinson (2010). Values and Assumptions Shaping Agile Development and User Experience Design in Practice. In A. Sillitti, A. Martin, X. Wang, E. Whitworth, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, and C. Szyperski (Eds.), *Agile Processes in Software Engineering and Extreme Programming*, Volume 48 of *Lecture Notes in Business Information Processing*, pp. 178–183. Springer Berlin Heidelberg. 88, 89, 90, 93, 94, 96, 98, 100, 101, 102, 129, 368

[85] Ferreira, J., H. Sharp, and H. Robinson (2011, August). User Experience Design and Agile Development: Managing Cooperation through Articulation Work. *Software Practice and Experience 41*(9), 963–974. 54, 88, 89, 93, 95, 96, 98, 100, 101, 102, 114, 115, 117, 120, 129, 318, 320, 368

[86] Ferreira, J., H. Sharp, and H. Robinson (2012). Agile Development and User Experience Design Integration as an On-going Achievement in Practice. In *Agile 2012*. 88, 93, 94, 96, 98, 100, 101, 102, 117, 129, 130, 320, 368, 435

[87] Flanagan, G. and T. Rauch (1995, October). Usability Management Maturity, Part 1: Self Assessment: How Do You Stack Up? *CHI 95 Conference Companion* (4). 53, 186, 187, 231, 283

[88] Fowler, M. (2005, December). The New Methodology. 18, 29, 402, 416

[89] Fox, D., J. Sillito, and F. Maurer (2008). Agile Methods and User-Centered

Design: How These Two Methodologies are Being Successfully Integrated in Industry. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 63–72. IEEE Computer Society. 54, 93, 94, 98, 100, 101, 102, 108, 109, 111, 112, 117, 121, 125, 128, 129, 159, 179, 302, 368

[90] Frank, A. and C. Hartel (2009). Feature Teams Collaboratively Building Products from READY to DONE. In *Proceedings of the 2009 Agile Conference*, AGILE '09, Washington, DC, USA, pp. 320–325. IEEE Computer Society. 88, 104

[91] Fuller, A. (2004). Usability/User Interface Design in Agile Processes. In *OZCHI*. 91, 368

[92] Garrett, J.-J. (2011). *The Elements of User Experience:User Centered Design for the Web and Beyond*. New Riders. 38, 408, 415, 416

[93] Ghanam, Y., X. Wang, and F. Maurer (2008). Utilizing Digital Tabletops in Collocated Agile Planning Meetings. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 51–62. IEEE Computer Society. 88

[94] Gottschalk, P. (2009). Maturity Levels for Interoperability in Digital Government. *Government Information Quarterly 26*, 75Ű–81. 56, 402

[95] Gould, J. and C. Lewis (1985, March). Designing for Usability: Key Principles and What Designers Think. *Communications of ACM 28*(3), 300–311. 18, 36, 37, 38, 39, 40, 109, 315, 408, 415, 416

[96] Greenhalgh, T. (2001). *How to Read a Paper?* (Second ed.). London: BMJ Publishing group. 76, 79

[97] Gulliksen, J., B. Goransson, I. Boivie, S. Blomkvist, J. Persson, and A. Cajander (2003). Key Principles for User-Centred Systems Design. *Behaviour and Information Technology 22*, 397–409. 35, 37, 38, 41, 406, 408, 414

[98] Gulliksen, J., A. Lantz, and I. Boivie (1999). User Centered Design in Practice-Problems and Possibilities. *Sweden: Royal Institute of Technology*. 315, 433

[99] Gulliksen, J., A. Lantz, and I. Boivie (2001). How to Make User Centred Design Usable. *SIGCHI Bulletin 33*(3), 16–31. 316, 320, 435

[100] Gundelsweiler, F., T. Memmel, and H. Reiterer (2004). Agile Usability Engineering. *Mensch and Computer*, 33–42. 54

[101] Gupta, A. (1997). Humanware Process Improvement Framework: Interfacing User Centred Design and the Product Creation Process at Philips. Position paper delivered at Human Centred Process Improvement Group (HCPIG) meeting.Teddington, UK. 187

[102] Haikara, J. (2007). Usability in Agile Software Development: Extending

the Interaction Design Process with Personas Approach. In *Proceedings of the 8th International Conference on Agile Processes in Software Engineering and Extreme Programming*, XP'07, Berlin, Heidelberg, pp. 153–156. Springer-Verlag. 32, 55, 56, 87, 88, 332

[103] Hakli, A. (2005). Introducing User Centered Design in a Small Size Software Developement Organization. Master's thesis, Helsinki University of Technology. 27, 405, 406, 407, 408, 414

[104] Hatinen, A. (2002). Extreme Programming and Goal Oriented User Interface Design in Practice. Research Seminar on Software Engineering -University of Helsinki-Department of Computer Science. 88

[105] Hauge, H. (2002). How to Combine Software Process Improvement and Quality Assurance with Extreme Programming. Master's thesis, IDI,NTNU. 18, 29, 402

[106] Helgesson, Y. Y. L., M. Host, and K. Weyns (2012). A Review of Methods for Evaluation of Maturity Models for Process Improvement. *Journal of Software: Evolution and Process 24*(4), 436–454. 340, 404

[107] Hellmann, T., A. Hosseini-Khayat, and F. Maurer (2011). Test-Driven Development of Graphical User Interfaces: A Pilot Evaluation. In *XP 2011*, pp. 223–237. 88

[108] Hellmann, T. and F. Maurer (2011, aug.). Rule-Based Exploratory Testing of Graphical User Interfaces. In *Agile Conference (AGILE)*, pp. 107 –116. 88

[109] Hellmann, T. D., A. Hosseini-Khayat, and F. Maurer (2010, april). Supporting Test-Driven Development of Graphical User Interfaces Using Agile Interaction Design. In *Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on*, pp. 444–447. 90

[110] Herbsleb, J., D. Zubrow, D. Goldenson, W. Hayes, and M. Paulk (1997, June). Software Quality and the Capability Maturity Model. *Communications of the ACM 40*(6), 30–40. 56, 402, 413, 416

[111] Hevner, A., S. March, J. Park, and S. Ram (2004). Design Science in Information Systems Research. *MIS Quarterly 28*, 75–105. 225, 328

[112] Highsmith, J. (2000). *Adaptive Software Developement: A Collaborative Approach to Managing Complex Systems*. Dorset House. 32

[113] Hodgetts, P. (2005). Experiences Integrating Sophisticated User Experience Design Practices into Agile Processes. In *Proceedings of the Agile Development Conference*, ADC '05, Washington, DC, USA, pp. 235–242. IEEE Computer Society. 54, 93, 94, 98, 100, 101, 102, 111, 112, 117, 119, 122, 302, 315, 370

[114] Holzinger, A., M. Errath, G. Searle, B. Thurnher, and W. Slany (2005). From Extreme Programming and Usability Engineering to Extreme Usability in Software Engineering Education (XP+UE→XU). In *Proceedings of the 29th Annual International Conference on Computer Software and Applications Conference*, COMPSAC-W'05, Washington, DC, USA, pp. 169–172. IEEE Computer Society. 54, 93, 94, 98, 99, 100, 101, 103, 126, 131, 133, 135, 368

[115] Hosseini-Khayat, A., Y. Ghanam, S. Park, and F. Maurer (2009). ActiveStory Enhanced: Low-Fidelity Prototyping and Wizard of Oz Usability Testing Tool. In P. Abrahamsson, M. Marchesi, F. Maurer, W. Aalst, J. Mylopoulos, M. Rosemann, M. J. Shaw, and C. Szyperski (Eds.), *Agile Processes in Software Engineering and Extreme Programming (XP2009)*, Volume 31 of *Lecture Notes in Business Information Processing*, pp. 257–258. Springer. 91, 368

[116] Hosseini-Khayat, A., T. Hellmann, and F. Maurer (2010, aug.). Distributed and Automated Usability Testing of Low-Fidelity Prototypes. In *Agile Conference (AGILE)*, pp. 59 –66. 90, 93, 98, 99, 100, 101, 102, 103, 135, 368

[117] Host, M. and P. Runeson (2007). Checklists for Software Engineering Case Study Research. In *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, ESEM '07, Washington, DC, USA, pp. 479–481. IEEE Computer Society. 76, 79

[118] Hudson, W. (2003). Adopting User-Centered Design within an Agile Process: A Conversation. *Cutter IT Journal 16.10*, 5–12. 91, 93, 95

[119] Hudson, W. (2005, January). A Tale of Two Tutorials: A Cognitive Approach to Interactive System Design and Interaction Design Meets Agility. *interactions 12*(1), 49–51. 90

[120] Humayoun, S. R., Y. Dubinsky, and T. Catarci (2011). A Three-Fold Integration Framework to Incorporate User-Centered Design into Agile Software Development. In *Proceedings of the 2nd International Conference on Human Centered Design*, HCD'11, Berlin, Heidelberg, pp. 55–64. Springer-Verlag. 93, 94, 98, 99, 100, 101, 102, 103, 135, 368

[121] Humphrey, W. (1997). *Introduction to the Personal Software Process*. Addison-Wesley. 29

[122] Humphrey, W. (2000). *Introduction to the Team Software Process*. Reading,Massachusetts.: Addison Wesley. 29

[123] Hussain, Z., M. Lechner, H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, T. Vlk, and P. Wolkerstorfer (2007). User Interface Design for a Content-aware Mobile Multimedia Application: An Iterative Approach. In *MoMM2007 & iiWAS2007 Workshops Proceedings*. 86, 90

[124] Hussain, Z., M. Lechner, H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, T. Vlk, and P. Wolkerstorfer (2008). User Interface Design for a Mobile Multimedia Application: An Iterative Approach. In *Proceedings of the First International Conference on Advances in Computer-Human Interaction*, ACHI '08, Washington, DC, USA, pp. 189–194. IEEE Computer Society. 54

[125] Hussain, Z., M. Lechner, H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, and P. Wolkerstorfer. Integrating Extreme Programming and User-Centered Design. In *PPIG*. 86, 90

[126] Hussain, Z., M. Lechner, H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, and P. Wolkerstorfer (2008). Agile User-Centered Design Applied to a Mobile Multimedia Streaming Application. In *Proceedings of the 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for Education and Work*, USAB '08, Berlin, Heidelberg, pp. 313–330. Springer-Verlag. 86, 90, 104

[127] Hussain, Z., M. Lechner, H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, and P. Wolkerstorfer (2009). Concept and Design of a Contextual Mobile Multimedia Content Usability Study. In *Proceedings of the 2009 Second International Conferences on Advances in Computer-Human Interactions*, ACHI '09, Washington, DC, USA, pp. 277–282. IEEE Computer Society. 86, 90

[128] Hussain, Z., H. Milchrahm, S. Shahzad, W. Slany, M. Tscheligi, and P. Wolkerstorfer (2009). Integration of Extreme Programming and User-Centered Design: Lessons Learned. In P. Abrahamsson, M. Marchesi, and F. Maurer (Eds.), *Agile Processes in Software Engineering and Extreme Programming*, Volume 31 of *Lecture Notes in Business Information Processing*, pp. 174–179. Springer Berlin Heidelberg. 86, 90, 109

[129] Hussain, Z., H. Milchrahm, S. Shahzad, W. Slany, M. Umgeher, T. Vlk, C. Koeffel, M. Tscheligi, and P. Wolkerstorfer (2012). Practical Usability in XP Software Development Processes. In *The Fifth International Conference on Advances in computer Human Interactions, ACHI 2012*. 86, 88, 90, 93, 94, 96, 98, 100, 101, 102, 104, 111, 112, 117, 119, 120, 122, 123, 124, 130, 131, 132, 170, 310, 313, 368, 432

[130] Hussain, Z., W. Slany, and A. Holzinger (2009a). Current State of Agile User-Centered Design: A Survey. In *Proceedings of the 5th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for e-Inclusion*, USAB '09, Berlin, Heidelberg, pp. 416–427. Springer-Verlag. 86, 90, 94, 103

[131] Hussain, Z., W. Slany, and A. Holzinger (2009b). Investigating Agile User-Centered Design in Practice: A Grounded Theory Perspective. In *Proceedings of the 5th Symposium of the Workgroup Human-Computer Interaction and Usability*

Engineering of the Austrian Computer Society on HCI and Usability for e-Inclusion, USAB '09, Berlin, Heidelberg, pp. 279–289. Springer-Verlag. 88, 93, 94, 96, 98, 100, 101, 102, 103, 104, 111, 121, 302, 368

[132] Hussain, Z., W. Slany, and A. Holzinger (2012). Agile Software Developement Methods and Usability/ User Centred Design: Prespectives from an Online Survey. *Journal of Systems and Software*. 54, 86, 88, 90, 93, 95, 96, 98, 100, 101, 102, 103, 104, 111, 124, 125, 128, 302, 368

[133] IBM. An Agile Aapproach to User Experience and Design. 88

[134] IEEE-Standard-1061 (1998). Software Quality Metrices Methodology. 36, 407

[135] Iivari, N. and T. Jokela (2001). Evaluating a Usability Capability Assessment. In *Proceedings of the Third International Conference on Product Focused Software Process Improvement*, PROFES '01, London, UK, UK, pp. 98–109. Springer-Verlag. 186

[136] Illmensee, T. and A. Muff (2009). 5 Users Every Friday: A Case Study in Applied Research. In *Proceedings of the 2009 Agile Conference*, AGILE '09, Washington, DC, USA, pp. 404–409. IEEE Computer Society. 93, 94, 98, 100, 101, 102, 111, 117, 119, 121, 122, 123, 124, 159, 179, 302, 310, 370, 430

[137] ISO9126, I. (1991). Software Product Evaluation:Quality Characteristics and Guidlines of their Use. 36, 407

[138] ISO/DIS9241-11, I. (1996). Guidance on Usability:Ergonomic Requirements for Office Work with Visual Display Terminals (VDT). 36, 37, 407

[139] ISO/IEC, I. (2000). 18529 Human Centred Lifecycle Process Descriptions. Technical report, ISO/IEC. 56

[140] Iversen, J., P. A. Nielsen, and J. Norbjerg (1999, March). Situated Assessment of Problems in Software Development. *ACM SIGMIS Database - Special Issue on Infomration Systems 30*(2), 66–81. 56, 58, 402, 413, 416

[141] John, B., L. Bass, R. Kazman, and E. Chen (2004). Identifying Gaps between HCI, Software Engineering, and Design, and Boundary Objects to Bridge Them. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '04, New York, NY, USA, pp. 1723–1724. ACM. 52

[142] Jokela, T. (2001a). An Assessment Approach for User-Centred Design Processes. In *EUROSPI*, Limerick. Limerick Institute of Technology Press. 187, 190

[143] Jokela, T. (2001b). *Assessment of User Centred Design Processes as a Basis for Improvement Action*. Ph. D. thesis, University of Oulu. 187, 190

[144] Jokela, T. (2002a). A Method-Independent Process Model of User-Centred Design. In *Proceedings of the IFIP 17th World Computer Congress - TC13 Stream on Usability: Gaining a Competitive Edge*, Deventer, The Netherlands, The Netherlands, pp. 23–38. Kluwer, B.V. 40, 41, 43, 187, 190

[145] Jokela, T. (2002b). Assessment of User-Centred Design Processes - Lessons Learnt and Conclusions. In *Proceedings of the 4th International Conference on Product Focused Software Process Improvement*, PROFES '02, London, UK, UK, pp. 232–246. Springer-Verlag. 187, 190

[146] Jokela, T. (2002c). Making User-Centred Design Common Sense: Striving for an Unambiguous and Communicative UCD Process Model. In *Proceedings of the Second Nordic Conference on Human-Computer Interaction*, NordiCHI '02, New York, NY, USA, pp. 19–26. ACM. 41, 187, 190, 296, 419

[147] Jokela, T. (2004). Evaluating the User-Centredness of Development Organisations: Conclusions and Implications from Empirical Usability Capability Maturity Assessments. *Interacting with Computers 16(6)*, 1095–113. 59, 187, 190, 301, 357, 425

[148] Jokela, T. (2005a). Certification of the User-Centredness of Development Organisation - A Way for Ensuring User Acceptance even before the Development of Software? In *International Research Workshop on User-driven IT Design and Quality Assurance*, Stockholm. 59

[149] Jokela, T. (2005b). Performance Rather than Capability Problems. Insights from Assessments of Usability Engineering Processes. In *Proceedings of the 6th international conference on Product Focused Software Process Improvement*, PROFES'05, Berlin, Heidelberg, pp. 115–127. Springer-Verlag. 36, 187, 190, 316, 433

[150] Jokela, T. (2008). Characterizations, Requirements, and Activities of User-Centered Design. The KESSU 2.2 Model. In E.-C. Law, E. Hvannberg, and G. Cockton (Eds.), *Maturing Usability*, Human-Computer Interaction Series, pp. 168–196. Springer London. 39, 187, 190

[151] Jokela, T. (2010). Usability Maturity Models: Making your Company User Centred. *User Experience Magazine 9:1*, 14–16. 59, 416

[152] Jokela, T. and P. Abrahamsson (2000). Modelling Usability Capability - Introducing the Dimensions. In *Proceedings of the Second International Conference on Product Focused Software Process Improvement*, PROFES '00, London, UK, UK, pp. 73–87. Springer-Verlag. 36, 37, 53, 59, 320, 407, 414, 416, 417

[153] Jokela, T. and P. Abrahamsson (2004). Usability Assessment of an Extreme Programming Project: Close Co-operation with the Customer Does Not Equal to Good Usability. In F. Bomarius and H. Iida (Eds.), *Product Focused Software*

Process Improvement, Volume 3009 of *Lecture Notes in Computer Science*, pp. 393–407. Springer Berlin / Heidelberg. 20, 39, 54, 55, 59, 93, 94, 98, 100, 102, 103, 135, 183, 320, 326, 368, 414, 416, 417, 435

[154] Jokela, T., M. Siponen, N. Hirasawa, and J. Earthy (2006). A Survey of Usability Capability Maturity Models: Implications for Practice and Research. *Behaviour and Information Technology 25(3)*, 263–282. 9, 59, 183, 185, 186, 187, 188, 292

[155] Jones, B. S. (1995). Total Systems Maturity.Internal Report.Version 2. Technical report, BAeSEMA, Glasgow. 231, 283

[156] Kane, D. (2003). Finding a Place for Discount Usability Engineering in Agile Development: Throwing Down the Gauntlet. In *Proceedings of the Conference on Agile Development*, ADC '03, Washington, DC, USA. IEEE Computer Society. 19, 50, 51, 54, 93, 94, 98, 100, 101, 103, 121, 122, 126, 159, 170, 179, 222, 368, 417

[157] Kautz, K. (2011). Investigating the Design Process: Participatory Design in Agile Software Development. *Information Technology and People 248:3*, 217–235. 88

[158] Kazman, R., L. Bass, and J. Bosch (2003, may). Bridging the Gaps between Software Engineering and Human-Computer Interaction. In *The International Conference on Software Engineering*, pp. 777 – 778. 52

[159] Kazman, R., L. Bass, and J. Bosch (2004). Bridging the Gaps II: Bridging the Gaps Between Software Engineering and human-computer interaction. In *The International Conference on Software Engineering*. 52

[160] King, J. L. and K. Kraemer (1984, May). Evolution and Organizational Information Systems: An Assessment of Nolan's Stage Model. *Communications of the ACM 27*(5), 466–475. 58, 224

[161] Kitchenham, B. and S. Charters (2007). Guidelines for Performing Systematic Literature Reviews in Software Engineering. Technical report, Keele University and Durham university Joint Report EBSE. 66, 67, 74, 75, 78, 81, 82

[162] Kitchenham, B., T. Dyba, and M. Jorgensen (2004). Evidence-Based Software Engineering. In *Proceedings of the 26th International Conference on Software Engineering*, ICSE '04, Washington, DC, USA, pp. 273–281. IEEE Computer Society. 66, 133, 194, 226, 291

[163] Kitchenham, B. and S. Pfleeger (2008). *Guide to Advanced Empirical Software Engineering* (First ed.). Springer London. 46, 48, 49, 140, 141, 200, 403, 404, 405, 406

[164] Kitchenham, B. A., S. L. Pfleeger, D. C. Hoaglin, K. E. Emam, and J. Rosenberg (2002, aug). Preliminary Guidelines for Empirical Research in Software

Engineering. *IEEE Transactions on Software Engineering 28*(8), 721 – 734. 75, 76, 79

[165] Kollmann, J. (2008). Designing the User Experience in an Agile Context. Master's thesis, University College London. 90, 91, 93, 96, 98, 100, 101, 102, 110, 111, 112, 115, 116, 117, 122, 127, 128, 131, 302, 307, 315, 317

[166] Kollmann, J., H. Sharp, and A. Blandford (2009, aug.). The Importance of Identity and Vision to User Experience Designers on Agile Projects. In *Agile Conference*, pp. 11–18. 54, 90

[167] Kotonya, G. and I. Sommerville (1998). *Requirements Engineering: Processes and Techniques*. Sussex, England.: John Wiley and Sons. 314, 433

[168] Krohn, T., M. C. Kindsmüller, and M. Herczeg (2009). User-Centered Design Meets Feature-Driven Development: An Integrating Approach for Developing the Web Application myPIM. In *Proceedings of the 1st International Conference on Human Centered Design: Held as Part of HCI International 2009*, HCD 09, Berlin, Heidelberg, pp. 739–748. Springer-Verlag. 32, 88, 332

[169] Kruchten, P. (2004). *The Rational Unified Process: An Introduction*. Addison Wesley. 29

[170] Kuutti, K., T. Jokela, M. Nieminen, and P. Jokela (1998, September). Assessing Human Centred Design Processes in Product Developement by Using the INUSE Maturity Model. In *Proceedings of the 7th Symposium on Analysis,Design and Evaluation of Man Machine Systems*. 192

[171] Larman, C. (2003). *Agile and Iterative Development: A Manager's Guide*. Addison-Wesley Professional. 32

[172] Larusdottir, M., E. Bjarnadottir, and J. Gulliksen (2010). The Focus on Usability in Testing Practices in Industry. In P. Forbrig, F. Patern?, and A. Mark Pejtersen (Eds.), *Human-Computer Interaction*, Volume 332 of *IFIP Advances in Information and Communication Technology*, pp. 98–109. Springer Boston. 93, 95, 98, 100, 102, 103, 122, 123, 315, 368

[173] Law, E., V. Roto, A. Vermeeren, J. Kort, and M. Hassenzahl (2008). Towards a Shared Definition of User Experience. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, New York, NY, USA, pp. 2395–2398. ACM. 37

[174] Lee, J. C. (2006). Embracing Agile Development of Usable Software Systems. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, New York, NY, USA, pp. 1767–1770. ACM. 89, 341

[175] Lee, J. C. (March 2010.). *Integrating Scenario-Based Usability Engineering*

and Agile Software Development. Ph. D. thesis, Department of Computer Science.Virginia Tech. 89, 96

[176] Lee, J. C., T. Judge, and S. McCrickard (2011). Evaluating EXtreme Scenario-Based Design in a Distributed Agile Team. In *Proceedings of the 2011 Annual Conference on Human factors in Computing Systems*, CHI EA '11, New York, NY, USA, pp. 863–877. ACM. 54, 89, 93, 94, 96, 98, 100, 101, 102, 117, 118, 129, 130, 319, 341, 368, 434

[177] Lee, J. C. and S. McCrickard (2007). Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development. In *Proceedings of the AGILE 2007*, AGILE '07, Washington, DC, USA, pp. 59–71. IEEE Computer Society. 89, 93, 94, 96, 98, 100, 101, 102, 106, 108, 109, 112, 116, 119, 120, 121, 122, 341, 368

[178] Lee, J. C., S. McCrickard, and T. Stevens (2009, Aug.). Examining the Foundations of Agile Usability with eXtreme Scenario-Based Design. In *Agile Conference*, pp. 3 –10. 19, 54, 55, 89, 93, 94, 96, 98, 100, 101, 102, 106, 108, 110, 113, 114, 115, 116, 117, 222, 308, 317, 341, 370, 417, 429, 433

[179] Leitner, M., P. Wolkerstorfer, R. Sefelin, and M. Tscheligi (2008). Mobile Multimedia: Identifying User Values Using the Means-End Theory. In *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '08, New York, NY, USA, pp. 167–175. ACM. 88

[180] Lester, C. (2011). Combining Agile Methods and User-Centered Design to Create a Unique User Experience: An Empirical Inquiry. In *ACHI 2011, The Fourth International Conference on Advances in Computer-Human Interactions*. 91, 368

[181] Leszek, A. and C. Courage (2008, Aug.). The Doctor is "In" – Using the Office Hours Concept to Make Limited Resources Most Effective. In *AGILE Conference 2008*, pp. 196 –201. 93, 94, 98, 100, 101, 102, 125, 126, 164, 179, 370

[182] Lewis, C. (1982). Using the Thinking Aloud Method In Cognitive Interface Design - Technical report RC-9265. Technical report, IBM. 48

[183] Lievesley, M. and J. Yee (2006). The Role of the Interaction Designer in an Agile Software Development Process. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, New York, NY, USA, pp. 1025–1030. ACM. 54, 93, 94, 98, 100, 102, 110, 113, 116, 129, 130, 308, 370

[184] Lindstrom, H. and M. Malmsten (2008). User-Centred Design and Agile Development: Rebuilding the Swedish National Union Catalogue. *The Code4Lib Journal 5*, 12–Ű15. 54, 93, 95, 98, 100, 101, 102, 103, 104, 112, 131, 298, 370

[185] Losada, B., M. Urretavizcaya, and I. Fernandez-deCastro (2011). Agile Development of Interactive Software by means of User Objectives. In *The Sixth International Conference on Software Engineering Advances*. 54, 93, 94, 98, 99, 368

[186] Maier, A., J. Moultrie, and J. Clarkson (2009). Developing Maturity Grids for Assessing Organisational Capabilities: Practitioner Guidance. In *4th International Conference on Management Consulting*, Vienna, Austria. 58, 192, 220, 221, 226

[187] March, S. and G. Smith (1995). Design and Natural Science Research on Information Technology. *Decision Support Systems 15*, 251–266. 328, 338, 349

[188] Martin, A. (2003). Exploring the XP Customer Role. In M. Marchesi and G. Succi (Eds.), *Extreme Programming and Agile Processes in Software Engineering*, Volume 2675 of *Lecture Notes in Computer Science*, pp. 427–428. Springer Berlin Heidelberg. 88

[189] Martin, A., R. Biddle, and J. Noble (2004). The XP Customer Role in Practice: Three Studies. In *Proceedings of the Agile Development Conference*, ADC '04, Washington, DC, USA, pp. 42–54. IEEE Computer Society. 88

[190] Martin, A., R. Biddle, and J. Noble (2009, aug.). The XP Customer Team: A Grounded Theory. In *Agile Conference 2009*, pp. 57 –64. 88

[191] Mayhew, D. (1999). *The Usability Engineering LifeCycle: A Practitioner Handbook for User Interface Design*. Morgan Kaufmann Publishers. 9, 46, 47, 407

[192] McClelland, I., T. Gelderen, B. Taylor, B. Hefley, and A. Gupta. Humanware Process Improvement - Institutionalising the Principles of User Centred Design. 187

[193] McInerney, P. and F. Maurer (2005, November). UCD in Agile Projects: Dream Team or Odd Couple? *Interactions 12*(6), 19–23. 54, 93, 95, 98, 100, 101, 102, 106, 107, 108, 110, 111, 113, 116, 121, 123, 128, 154, 170, 173, 302, 305, 317, 368

[194] McNeil, M. (2005). User Centred Design in Agile Application Developement. 88

[195] Mcneil, M. (2006). Agile User-Centred Design. In *Proceedings of the International Conference on Contemporary Ergonomics (CE2006)*, Cambridge, UK. 88, 91, 93, 94, 98, 100, 101, 103, 106, 109, 110, 111, 112

[196] Medlock, M., D. Wixon, M. Terrano, R. Romero, and B. Fulton (2002). Using the RITE method to improve products: A definition and a case study. In *Usability Professionals Association*. 51

[197] Memmel, T. (2006). Agile Usability Engineering. 54, 88

[198] Memmel, T., F. Gundelsweiler, and H. Reiterer (2007). Agile Human-Centered Software Engineering. In *Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI...but not as we know it - Volume 1*, BCS-HCI '07, Swinton, UK, UK, pp. 167–175. British Computer Society. 93, 94, 98, 100, 101, 103, 121, 122, 126, 128, 159, 179, 368

[199] Memmel, T., H. Reiterer, and A. Holzinger (2007). Agile Methods and Visual Specification in Software Development: A Chance to Ensure Universal Access. In *Proceedings of the 4th International Conference on Universal Access in Human Computer Interaction: Coping with Diversity*, UAHCI'07, Berlin, Heidelberg, pp. 453–462. Springer-Verlag. 93, 94, 98, 100, 101, 102, 104, 112, 116, 121, 308, 368, 429

[200] Meszaros, G. and J. Aston (2006). Adding Usability Testing to an Agile Project. In *Proceedings of the conference on AGILE 2006*, AGILE '06, Washington, DC, USA, pp. 289–294. IEEE Computer Society. 93, 94, 98, 100, 101, 102, 103, 110, 112, 121, 370

[201] Mettler, T. (2011). Maturity Assessment Models: A Design Science Research Approach. *International Journal of Society Systems Science 3*, 81–98. 56, 58, 60, 61, 62, 203, 328, 331, 333, 334, 335, 402, 413, 416, 419

[202] Mettler, T. and P. Rohner (2009). Situational Maturity Models as Instrumental Artifacts for Organizational Design. In *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*, DESRIST '09, New York, NY, USA, pp. 22:1–22:9. ACM. 58

[203] Miller, L. (2005). Case Study of Customer Input For a Successful Product. In *Proceedings of the Agile Development Conference*, ADC '05, Washington, DC, USA, pp. 225–234. IEEE Computer Society. 39, 54, 93, 94, 98, 100, 101, 102, 111, 112, 115, 116, 117, 118, 119, 168, 169, 302, 306, 307, 308, 370, 425, 428

[204] Miller, L. and D. Sy (2009). Agile User Experience SIG. In *Proceedings of the 27th International Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, New York, NY, USA, pp. 2751–2754. ACM. 88, 98, 100, 101, 103, 110, 113, 123, 130, 131, 132, 173, 308, 309, 313, 432

[205] Moallem, A. (2009). Design and Improve User Experience in Agile Developement Cycles. 87, 88

[206] Moreno, A. and A. Yagie (2012). Agile User Stories Enriched with Usability. In C. Wohlin (Ed.), *Agile Processes in Software Engineering and Extreme Programming*, Volume 111 of *Lecture Notes in Business Information Processing*, pp. 168–176. Springer Berlin Heidelberg. 93, 94, 98, 99, 100, 101, 103, 126, 127, 128, 135, 305, 368

[207] Moreno, A. and A. Yague (2010). First XP Workshop about Dealing with Usability in an Agile Domain. 54, 88

[208] Moreno, A. and A. Yague (2011). Second XP Workshop about Dealing with Usability in an Agile Domain. 54, 88

[209] Najafi, M. and L. Toyoshiba (2008). Two Case Studies of User Experience Design and Agile Development. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 531–536. IEEE Computer Society. 93, 94, 98, 100, 101, 102, 111, 112, 116, 117, 118, 119, 129, 133, 136, 226, 291, 302, 310, 318, 320, 370, 430

[210] Neilsen, J. (2006). Jakon Neilsen's Alertbox: Corporate Usability Maturity:Stages 5-8. 347

[211] Neilsen, J. and R. Mack (1994). *Usability Inspection Methods*. John Wiley & Sons. 50, 51, 403

[212] Nelson, E. (2002). Extreme programming vs. Interaction Design. 88

[213] Nielsen, J. (1989). Usability Engineering at a Discount. In *Proceedings of the Third International Conference on Human-Computer Interaction on Designing and Using Human-Computer Interfaces and Knowledge Based Systems (2nd ed.)*, New York, NY, USA, pp. 394–401. Elsevier Science Inc. 50, 403

[214] Nielsen, J. (1993). *Usability Engineering*. San Francisco, USA: Morgan Kaufmann. 37, 39, 45, 120, 407

[215] Nielsen, J. (2006). Jakon Neilsen's Alertbox: Corporate Usability Maturity:Stages 1-4. 12, 23, 186, 204, 225, 347

[216] Nielsen, J. and V. Phillips (1993). Estimating the Relative Usability of Two Interfaces: Heuristic, Formal, and Empirical Methods Compared. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, New York, NY, USA, pp. 214–221. ACM. 36, 37, 407

[217] Nodder, C. and M. Federoff (2008). Agile Developement and Usability. 88

[218] Nodder, C. and J. Nielsen (2009). Agile Usability: Best Practice for User Experience on Agile Developement Projects. http://www.nngroup.com/reports/agile. 88

[219] Norman, D. (2002). *The Design of Everyday Things*. MIT Press. 18, 38, 408

[220] Nummiaho, A. (2006). User-Centered Design and Extreme Programming. 87, 88

[221] Obendorf, H. and M. Finck (2008). Scenario-Based Usability Engineering Techniques in Agile Development Processes. In *CHI '08 Extended Abstracts on*

Human Factors in Computing Systems, CHI EA '08, New York, NY, USA, pp. 2159–2166. ACM. 54, 55, 89, 93, 94, 98, 100, 101, 102, 106, 110, 111, 112, 128, 368

[222] Obendorf, H., A. Schmolitzky, and M. Finck (2006). XPnUE Ű Defining and Teaching a Fusion of Extreme Programming and Usability Engineering. In *HCI Educators Workshop 2006 Inventivity: Teaching Theory, Design and Innovation in HCI*. 55, 89

[223] Ohlhauser, J. (Ohlhauser-Agile-2008). Experience Report: Design to Delivery in 7 Weeks. In *Agile Conference 2008*. 88

[224] Packard, H. (2007). The HP Business Intelligence Maturity Model. 56

[225] Palmer, S. and J. Felsing (2002). *A Practical Guide to Feature Driven Developement*. Prentice Hall. 32, 55

[226] Parsons, D., R. Lal, and H. Ryu (2007). Software Developement Methodologies, Agile Developement and Usability Engineering. In *18th Australian Conference on Information Systems*. 93, 94, 98, 100, 101, 102, 370

[227] Patton, J. (2002a). Designing Requirements: Incorporating Usage-Centered Design into an Agile SW Development Process. In D. Wells and L. Williams (Eds.), *Extreme Programming and Agile Methods Ů XP/Agile Universe 2002*, Volume 2418 of *Lecture Notes in Computer Science*, pp. 95–102. Springer Berlin / Heidelberg. 89, 103, 104

[228] Patton, J. (2002b). Hitting the Target: Adding Interaction Design to Agile Software Development. In *OOPSLA 2002 Practitioners Reports*, OOPSLA '02, New York, NY, USA. ACM. 89, 93, 94, 98, 99, 100, 101, 103, 104, 112, 133, 369, 370

[229] Patton, J. (2003). Improving on Agility: Adding Usage Centred Design to Typical Agile Software Development Environment. In *Second International Conference on Usage-Centered Design*. 89

[230] Patton, J. (2004). Interaction Design Meets Agility: Practicing Usage Centered Design on Agile Development Projects. In C. Zannier, H. Erdogmus, and L. Lindstrom (Eds.), *Extreme Programming and Agile Methods - XP/Agile Universe 2004*, Volume 3134 of *Lecture Notes in Computer Science*, pp. 35–64. Springer Berlin / Heidelberg. 54, 88

[231] Patton, J. (2005). Finding the Forest in the Trees. In *Companion to the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, OOPSLA '05, New York, NY, USA, pp. 266–274. ACM. 54, 104

[232] Patton, J. (2008a). Bringing User-Centered Design Practices Into Agile Development Projects. 54, 88

[233] Patton, J. (2008b). Twelve Emerging Best Practices for Adding UX Work to Agile Development. 88

[234] Patton, J. and T. Rauch (2005). Full Day Tutorial: Agile User Centered Design: Practicing UCD on Agile Projects. UPA. 54, 88

[235] Paulk, M., B. Curtis, M. Chrissis, and C. Weber (1993). Capability Maturity ModelSM for Software, Version 1.1. Technical report, Software Engineering Institute.Carnegie Mellon University. 56

[236] PC, A. and B. Prabhu (2012, june). Integrating Requirements Engineering and User Experience Design in Product Life Cycle Management. In *First International Workshop on Usability and Accessibility Focused Requirements Engineering (UsARE)*, pp. 12 –17. 88

[237] Pearce, L. (2005). *How to Examine a Thesis*. New York: Open University Press. 227

[238] Peixoto, C. S. A. (2009, june). Human-Computer Interface Expert System for Agile Methods. In *International Conference on Information Technology Interfaces*, pp. 311–316. 94

[239] Peixoto, C. S. A. and A. E. A. da Silva (2009). A Conceptual Knowledge Base Representation for Agile Design of Human-Computer Interface. In *Proceedings of the 3rd international Conference on Intelligent Information Technology Application*, IITA'09, Piscataway, NJ, USA, pp. 156–160. IEEE Press. 93, 94, 97, 98, 100, 102, 103, 368

[240] Poppelbub, J. and M. Roglinger (2011). What Makes a Useful Maturity model? A Framework of General Design Principles for Maturity Models and its Demonstration in Business Process Management. In *European Conference on Information Systems*. 224, 226, 285, 288, 419

[241] Preece, J., Y. Rogers, and H. Sharp (2002). *Interaction Design: Beyond Human Computer Interaction*. John Wiley & Sons. 18, 35, 36, 37, 38, 40, 44, 45, 46, 48, 49, 50, 51, 120, 299, 403, 404, 405, 406, 407, 408, 414, 415

[242] Raithatha, D. (2007). Making the Whole Product Agile Ű A Product Owners Perspective. In *XP*. 88

[243] Rannikko, P. (2011, April). User Centred Design in Agile Software Developement. Master's thesis, University of Tampere, School of Information Science. 91, 93, 96, 98, 100, 101, 103

[244] Rauch, T. and G. Flanagan (1995). Usability Management Maturity, Part 2: Usability Techniques - What Can You Do? In *CHI 95 Conference Companion*. 186, 187

[245] Raza, A., L.-F. Capretz, and F. Ahmed (2012, July). An Open Source Usability Maturity Model (OS-UMM). *Computers in Human Behavior 28*(4), 1109–1121. 188

[246] Rittenbruch, M., G. McEwan, nigel Ward, T. Mansfield, and D. Bartenstein (2002). Extreme Participation - Moving Extreme Programming Towards Participatory Design. In *Proceedings of the Seventh Biennial Participatory Design Conference*. 93, 94, 98, 100, 101, 102, 130, 313, 314, 315, 368

[247] Robson, C. (2011). *Real World Research: A Resource for Social Scientists and Practitioner-Researchers*. Blackwell Publishers Ltd. 140, 200

[248] Rosemann, M. and T. deBruin (2005). Towards a Business Process Management Maturity Model. In *European Conference on Information Systems*, Regensburg. Germany. 56, 402, 413, 416

[249] Rosenbaum, S., J.-A. Rohn, and J. Humburg (2000). A Toolkit for Strategic Usability: Results from Workshops, Panels, and Surveys. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, New York, NY, USA, pp. 337–344. ACM. 53

[250] Rosson, M.-B. and J. Carroll (2002). *Usability Engineering: Scenario-Based Developement of Human Computer Interaction*. Morgan Kaufmann.New York. 45, 405

[251] Salah, D. (2011, may). A Framework for the Integration of User Centered Design and Agile Software Development Processes. In *33rd International Conference on Software Engineering (ICSE)*, pp. 1132–1133. 54, 91, 93, 94, 98, 100, 102, 103

[252] Salah, D. (2013, February). Agile and user centred design integration workshop-what lies beneath and what lies ahead (aucdi 2013). 54

[253] Salah, D., R. Paige, and H. Petrie (2010). Practices in the Integration of Agile and UCD Processes- A Reality Check. In *Agile 2010.UX Session.*, Nashville, Tennessee. 88

[254] Salah, D., H. Petrie, and R. Paige (2009). Towards a Framework for Bridging User-Centred Design and Agile Software Development Processes. In *3rd Irish HCI Conference 2009*. 93, 94, 98, 100, 101, 106, 107, 368

[255] Schwaber, K. (1995). SCRUM Development Process. In *Proceedings of the 10th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA)*, pp. 117–134. 32, 33

[256] Schwaber, K. (2004). *Agile Project Management with Scrum*. Redmond, WA.: Microsoft Press. 32, 33

[257] Schwaber, K. and M. Beedle (2001). *Agile Software Development with Scrum*. Upper Saddle River, NJ.: Prentice Hall PTR. 32, 33, 34, 108

[258] Seffah, A., J. Gulliksen, M. Desmarais, B. Goransson, I. Boivie, J. Persson, S. Blomkvist, E. Metzker, B. Jerome, R. Kazman, A. Sutcliffe, R. Adams, L. Bass, B. John, S. Kujala, T. Jokela, J. Carter, J. Liu, K. Schneider, D. Fourney, and X. Ferre (2005). *Human Centred Software Engineering-Integrating Usability in the Development Process*, Volume 2005 of *Human-Computer Interaction Series*. Springer Netherlands. 52, 53

[259] Seffah, A. and E. Metzker (2004, December). The Obstacles and Myths of Usability and Software Engineering. *Communications of the ACM 47*(12), 71–76. 35, 36, 37, 53, 407, 414

[260] Sharp, H., R. Biddle, P. Gray, L. Miller, and J. Patton (2006). Agile Development: Opportunity or Fad? In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, New York, NY, USA, pp. 32–35. ACM. 54, 88

[261] Sharp, H., H. Robinson, and J. Segal (2004). Integrating User-Centred Design and Software Engineering: a Role for eXtreme Programming? In *BCS-HCI Group's 7th Educators Workshop: Effective Teaching and Training in HCI*. 54, 87, 91, 93, 94, 98, 99, 102, 103, 106, 108, 109, 110, 130, 131, 161, 313, 314, 321, 368, 433, 435

[262] Singh, M. (2008, aug.). U-SCRUM: An Agile Methodology for Promoting Usability. In *Agile Conference*, pp. 555 –560. 54, 93, 94, 98, 100, 101, 102, 103, 104, 110, 111, 112, 113, 126, 179, 308, 309, 341, 370

[263] SJ, T. and B. R (1984). *Introduction to Qualitative Research Methods*. Wiley.New York. 141

[264] Sjoberg, D., T. Dyba, and M. Jorgensen (2007). The Future of Empirical Methods in Software Engineering Research. In *2007 Future of Software Engineering*, FOSE '07, Washington, DC, USA, pp. 358–378. IEEE Computer Society. 72, 75, 98, 133

[265] Skov, M. and J. Stage (2005). Supporting Problem Identification in Usability Evaluations. In *Proceedings of the 17th Australia Conference on Computer-Human Interaction: Citizens Online: Considerations for Today and the Future*, OZCHI '05, Narrabundah, Australia, Australia, pp. 1–9. Computer-Human Interaction Special Interest Group (CHISIG) of Australia. 36

[266] Smith, T. (2008, aug.). Product Innovation is Practical, Important, and Possible. In *AGILE 2008 Conference*, pp. 561–565. 88

[267] Sohaib, O. and K. Khan (2010, june). Integrating Usability Engineering and Agile Software Development: A Literature Review. In *Computer Design and*

Applications (ICCDA), 2010 International Conference on, Volume 2, pp. V2–32 – V2–38. 19, 54, 55, 67, 93, 94, 98, 100, 103, 107, 222, 368

[268] Stapelton, J. (1997). *Dynamic Systems Developement Method -The Method in Practice*. Addison Wesley. 32, 55

[269] Stapelton, J. (2003). *DSDM: Business Focused Development*. Pearson Education. 32, 55

[270] Sun (2005). Information Life Cycle Management Maturity Model. 56

[271] Sy, D. (2006). Formative Usability Investigations for Open-Ended Tasks. 54, 88

[272] Sy, D. (May 2007). Adapting Usability Investigations for Agile User-Centered Design. *Journal of Usability Studies 2(3)*, 112–132. 54, 93, 95, 98, 100, 101, 102, 103, 107, 110, 111, 112, 117, 128, 173, 302, 305, 308, 370, 429

[273] Sy, D. and L. Miller (2008). Optimizing Agile User-Centred Design. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, New York, NY, USA, pp. 3897–3900. ACM. 54, 88, 91, 93, 94, 98, 100, 101, 103, 129, 130

[274] Toxboe, A. (2005, May). Introducing User-Centered Design to eXtreme Programming. 88

[275] Tzanidou, K. and J. Ferreira (2010). Design and Development in the ŞAgile RoomŤ: Trialing Scrum at a Digital Agency. In A. Sillitti, A. Martin, X. Wang, and E. Whitworth (Eds.), *Agile Processes in Software Engineering and Extreme Programming*, Volume 48 of *Lecture Notes in Business Information Processing*, pp. 372–378. Springer Berlin Heidelberg. 93, 94, 98, 100, 102, 112, 128, 129, 370

[276] Uldall-Espersen, T., E. Frokjaer, A. Blandford, and T. Jokela (2007). Increasing the Impact of Usability Work in Software Development. In *CHI '07 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '07, New York, NY, USA, pp. 2873–2876. ACM. 52, 297

[277] Ungar, J. (2008). The Design Studio: Interface Design for Agile Teams. In *Proceedings of the Agile 2008*, AGILE '08, Washington, DC, USA, pp. 519–524. IEEE Computer Society. 54, 89, 110, 111, 115, 116, 117, 153, 179, 308, 317, 320, 341, 429, 433, 435

[278] Ungar, J. and J. White (2008). Agile User Centered Design: Enter the Design Studio - A Case Study. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, New York, NY, USA, pp. 2167–2178. ACM. 54, 93, 94, 98, 100, 101, 102, 179, 341, 370

[279] Venturi, G. and J. Troost (2004). Survey on the UCD Integration in the In-

dustry. In *Proceedings of the Third Nordic Conference on Human-Computer Interaction*, NordiCHI '04, New York, NY, USA, pp. 449–452. ACM. 52, 178

[280] Visconti, M. and C. Cook (1998, August). Evolution of a Maturity Model : Critical Evaluation and Lessons Learned. *Software Quality Control 7*(3/4), 223–237. 347

[281] Wang, X. and F. Maurer (2008, oct.). Tabletop AgilePlanner: A Tabletop-Based Project Planning Tool for Agile Software Development Teams. In *3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems.TABLETOP 2008.*, pp. 121 –128. 88

[282] Wells, D. (2009, September). Extreme Programming: A Gentle Introduction. 9, 33

[283] Williams, H. and A. Ferguson (2007). The UCD Perspective: Before and After Agile. In *Proceedings of the AGILE 2007*, AGILE '07, Washington, DC, USA, pp. 285–290. IEEE Computer Society. 87, 93, 94, 98, 100, 101, 102, 111, 112, 122, 126, 128, 153, 164, 179, 302, 370

[284] Wolkerstorfer, P., M. Tscheligi, C. Oppenauer-Meerskraut, and A. Geven (2010). Matching HCI Methods and Developers Values in eXtreme Programming Development Processes. In *I-UxSED 2010*, pp. 28–29. 91, 368

[285] Wolkerstorfer, P., M. Tscheligi, R. Sefelin, H. Milchrahm, Z. Hussain, M. Lechner, and S. Shahzad (2008). Probing an Agile Usability Process. In *CHI '08 Extended Aabstracts on Human Factors in Computing Systems*, CHI EA '08, New York, NY, USA, pp. 2151–2158. ACM. 86, 90, 106, 107

[286] Yin, R. (2009). *Case Study Research: Design and Methods*. Beverly Hills; London: SAGE. 72, 73, 193, 226, 291