COMBINING AGILE AND HUMAN CENTERED DESIGN APPROACH FOR USABILITY TESTING

ERIC NDIZIHIWE

MASTERS OF SCIENCE

(Software Engineering)

JOMO KENYATTA UNIVERSITY OF AGRICULTURE AND TECHNOLOGY

2017

Combining agile and human centered design approach for usability testing

ERIC NDIZIHIWE

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Software Engineering in the Jomo Kenyatta University of Agriculture and Technology

2017

DECLARATION

This thesis is my original work and has not been presented for a degree in any other University.

Signed: ……. Date: ……

Eric Ndizihiwe

This thesis has been submitted for examination with our approval as the university supervisors

Signed: ……..……. Date: ……

Prof. Wilson Kipruto Cheruiyot

JKUAT, Kenya

Signed: ……..……. Date: ……

Prof. Stephen Kimani

JKUAT, Kenya

DEDICATION

This research is dedicated to my sisters Aline Nyiranshuti and Joselyne Ingabire, whose love, care, support and encouragement made it possible for me to pursue a Master's of Science in Software Engineering. Likewise, the research's dedication goes to my old friends Elvis Shema, Laurien Ngirimana, Julien Simbi, Jean Paul Turikumwe, Yves Benimana, and Jean Marie Vianny Bikorimana. God has blessed me with the kindest people.

ACKNOWLEDGEMENTS

I would like to thank; first and foremost, to the whole team of Jomo Kenyatta University of Agriculture and Technology Kigali Campus (JKUAT) for the wonderful work they have done in my favor. I would like to thank also my supervisors Prof. Wilson Cheruiyot and Prof. Stephen Kimani for their support and encouragement throughout this research, and I would also like to thank my sisters and friends who are big important in my life.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CPU: Central ProcessinLg Unit

ERP: Enterprise Resources Planning

GUIs: Graphical User Interfaces

HCD: Human Centered Design

HCI: Human Computer Interaction

IT: Information Technology

ISO: International Organization for Standardization

RE: Requirement Engineering

SE: Software Engineering

UCD: User Centered Design

UI: User Interface

UX: User Experience

XP: eXtreme Programming

ABSTRACT

Agile was proposed by user interface developers as the faster and more effective way on how to transform user needs into the final system. The goal of this research was to seek how to integrate usability and interaction design methods into agile development process by using human centered design approach. To conduct the research efficiently and successfully, the prototype model was developed using agile and human centered design approach and tested. The research used comparative method to determine the different kind of usability results for software developed in agile and prototype model which was developed using agile with human centered design. As well, the data for this research was obtained from usability testing tool entitled "load complete". The research results showed the developed model counted 26% for total load page time compared with agile model which it counted 74%, the results showed that response time took 19% for developed model where agile model took 81%, by comparing that response time was high for agile software as compared with developed model. The results also showed that agile model used 99% CPU usage more than developed model which it used 68% CPU usage. The results demonstrated that the developed model can improve the efficiency and usability of interaction in software development process.

CHAPTER ONE

INTRODUCTION

1.1 Background

Software development is becoming the essential aspect to improve system facilities, maintain a focus on the rapid delivery of business value and satisfy user's needs (Saurav, 2015). As the field in which software development process is a tryout to construct and standardize development to produce the outcome of a project desirable and needed.Agile methodology was created by practitioners in the software industry and the goal was on qualifying software developers to increase production of software that meets customer needs.

1.1.1 Software Development processes

Software development process is a set of steps that a software program goes through when developed (Nielsen, 2011). There are many phases available and being used including requirements, design, and implementation, testing validation, documentation and maintenance.

The option of the appropriate process depends on various factors such as risk level, requirements stability, time-to-market and etc. The clearer the requirements and the more stringent the organizational structure are, the more structured and heavy weight the process should be (Hussain el, 2012).

1.1.2 Usability Testing

Usability testing describes area where real users have problems with application and helps users make proposition for improvement. The goal of usability testing is to check theuser interface and human factors applications with the people who will be interacting with the platform.

1.1.3

1.1.4 Agile Software Development

Agile software development is presently an emerging discipline in Software Engineering (SE) field.According to (Kuda et al., 2011) customers found that it difficult to define their needs because of changing software technologies within a short time of period often.

Agile as new methods are being designed to define the changing of user requirement in Software Development environment.The Agile was proposed by developers as the faster and more effective way to develop software that accurately meet customer satisfaction by structuring the development process into iterations.

Agile methods focus on the challenges of unpredictability of the real world by relying on the end people and their creativity.Agile values stated in the Agile Manifesto in February 2001, outline the ideas of agile development.

The principles behind the Agile Manifesto aims to help software developers to succeed in changing and unpredictable environments, promote and speed up responses to changing environment, requirement and meeting deadlines.

What is new about agile methods is not the practices they use, but their recognition of people as primary driver's project success, coupled with an intense focus on effectiveness and maneuverability (Abrahamsson, Oza, & Siponen, 2010).

Agile Manifesto states the focus of the agile development as the following:

(i) **Individuals and interactions** are valued over processes and tools
(ii) **Working software is** valued over comprehensive documentation
(iii) **Customer collaboration** is valued over contract negotiation
(iv) **Responding to change** is valued over following a plan

Agile approach strives to deliver better software by involving the customer representative closely throughout the development process and by delivering software product in small iterations as quickly as possible. However, it did not consider the time for user's research, experience design or usability testing. There has been some evidence that agile software development practices alone do not always ensure that they build software that is usable and what the end user needs (user requirements). Even if it may have improved the development process, it still produces some poor user's experiences (Jim, 2014).

1.1.5 Human Centered Design

Human Centered Design (HCD) has focused on the interactivity of the system. It has also been applied in development process functionality. However, the principles and objective of HCD could also be used in the agile implementation process.

User's experience is the extent to which specific users can use a software product to their satisfaction in order to effectively and efficiently achieve specific goals in a specific context of use. User's experience is defined as the perceptions and responses of users that result from their experience of using a product (Garret, 2011).

The benefit of usability improves productivity and raises team morale, reduces training and documentation cost.

According to (Xavier & Natalie, 2001) usability is usually divided into the following attributes:

(i) **Learnability**: How easy it is to learn the main system functionality.

(ii) **Efficiency**: The number of tasks per unit of time that the user can perform using the system.

(iii) **User retention over time**: It is critical for intermittent users to be able to use the system. It reflects how well the user can remember theworking process of the system after a certain period of time without using it.

(iv) **Error rate**: It addresses the number of errors the user makes while performing a task, this reduces efficiency and user satisfaction.

(v) **Satisfaction**: It shows a user's subjective impression of the system.

Usability is still insufficient in most products which developed in agile methodology; the most widespread view in software engineering is that usability is chiefly related to the user's interface. If the user's task is not properly supported, the user's needs (requirement) are not met and the objective of building the software expectation is missed.

The integration of Agile and User Experience Design are bound to the interesting performance of agile methods to quickly provide system products that answer the users' needs, efficiency and satisfaction with a certain level of quality.

It makes sense to build a user interface (UI) that improves the experience of using it in a positive and productive manner for the user.

Agile methods concentrate on expeditious delivery of working software to the customer with minimal upfront design, whereas User Experience Design tends to allocate more time and resources for research and testing before a single line of code is written. In agile methodology, working with software is delivered to the users in an iterative manner and for better feedback, the design of the system must be good so that the user owns the product and are also happy with that. Human Centered Design focuses on the customer software product design that the user wants.

The aim of this research was to explore ways of combining elements of HCD and agile methods in ways that maintain the benefits of both approaches. In doing so, this research

provided further valuable information that will be helpful to the teams in the software industry interested in combining these approaches in their current process.

Most importantly, agile development and Human centered design are both fueled by being highly iterative. The creation of an artifact can be validated and refined based on a feedback gathered.

This research sought to understand the close relationship between the two methodologies and how they can be used together for software development projects that yield to high levels of user's involvement while creating an enriched user and developer experience.

1.2 Statement of the problem

Agile methodology was focused on iterative development, emphasizes communication and collaboration of self-organized teams over comprehensive documentation and flexibility to changing requirements into software that meets customer needs. However, agile development does notfocuson usability. In the short development cycle, it does not consider the users’needs. In other words, Agile doesn't include user evaluation and it has limited customer interaction during the process of creating software.As an example,the requirements are provided literally by stakeholders and frequently people who perform as user representatives.This is not practically the same as considering the requirement provided bythe end-users.

According to Jim (2014), in the original conception of agile development, there was no user experience design role. Development began directly from the requirements.

It resulted in developers "designing" the user interface during coding, which often produced a poor user experience.

1.3 Justification

The justification of this research was to describe the challenges faced in Agile development process, to prove how it would be necessary to evolve the role of usability in order to help incorporate User Centered Design practices into software development process and implementations by using Agile methodology.

1.4 Objectives

1.4.1 General Objective

The general objective was to develop a model that combine Agile and Human Centered Design (HCD) approach for usability testing.

1.4.2 Specific Objective

The specific objective of this research was to:

(i) Analyze techniques for usability testing in development process for software.

(ii) Develop a model that combines agile and HCD for usability testing that improves efficientand interactive software development.

(iii) Evaluate the developed model using combined agile and HCD for usability testing.

1.5 Research Questions

(i) What are techniques used in software development process for testing usability?

(ii) What is the best approach for developing a model that combines agile and HCD for usability testing that improves efficient and interactive software development?

(iii) How can the proposed model be evaluated?

1.6 Scope of Research

The research sought to fully understand the usability testing in development process for software, analyze how Agile and Human Centered Design techniques can help to improve usability for user interfaces and user needs.

1.7 Organization of the research

In addition to the Introduction chapter, the organization of the research is divided into the following chapters:

Chapter two–this chapter discussesthe literature review. The chapter presents the software development process, agile methodology, human centered design (HCD) methods, agile methodology combined with human centered design, the summary and research gaps.

Chapter three – this chapter describes the research methodology used in this research including how data was collected and analyzed. The chapter also describes the proposed framework and testing tool used.

Chapter four –this chapter presents the analysis results and discussion of the research.

Chapter five – this chapter summarizes the research conclusion and recommendations for future work.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

Recent researchers have examined the integration of user centered design (UCD) with agile methods to determine if and how these two approaches can cohabitate on the same software project for usability purpose.

2.2Software Development Process

Software development process provides a series of steps to be followed to design and develop a software product efficiently (Nielsen, 2011). It is often studied that a subset of the systems development life cycle, the methodology may consist of pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application.

The selection of model has very high impact on the testing that is carried out. It will define the what, where and when of our planned testing, influence regression testing and largely determines which test techniques to use.

There are various Software development models or methodologies. They are as follows:

(i)Waterfall model: all phases will function one after another, when the first phase is finished then only the second phase will start and so on; (ii) Iterative model is repeating every step after every cycle of software development process; (iii) This model of development combines the features of the prototyping model and the waterfall model; (iv) V-Model provides means of testing of software at each stage in a reverse manner;(v) prototype model (Agile model).

It is very important to choose the right mode in software development process. Nowadays, on the market, the agile methodology is the most used model (Jim, 2014).

2.3Agile Development

Developing good software requires some form of methodology to ensure that the process of developing that software is standard comparing with others.

There have been many studies and suggestion for improving the development process. The new software development method called agile is an alternative to traditional approaches to developa software, aimed to focus on developing solutions more quickly and satisfying the customer (Fowler, 2005).

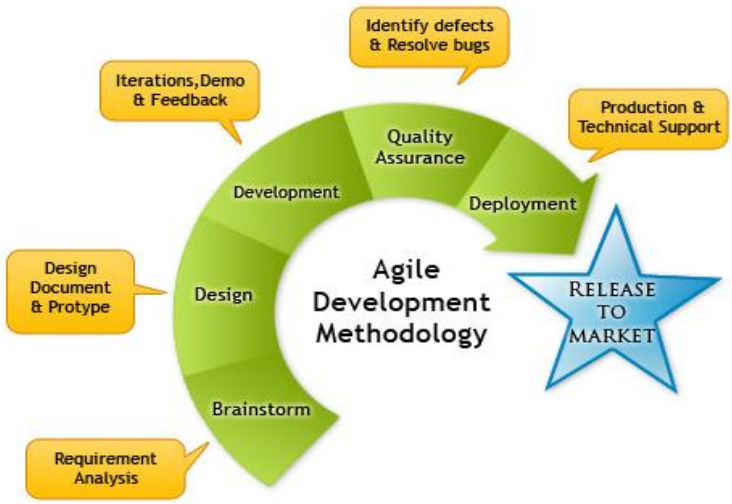

Figure 2.1: Agile Development process(Adopted from Silva, 2013).

The priority of Agile is customer satisfaction, which is accomplished by iterative deliveries of small working sets of features to the customer (Silva et al., 2013). This focus on functionality can come at the cost of usability, as Agile places emphasizes on minimal up-front design work.

Agile values stated in the Agile Manifesto in 2001, outline the ideas of agile development.

The principle behind the Agile Manifesto aims to help software developers to succeed in changing and unpredictable environments. The main theme in agile methods is to promote and speed up responses to changing environments, requirement and meeting deadlines.

Access to end users can be complex and difficult when dealing with any single service but it can be even more complex with multi service programs. Agile developers need to have a single voice for the user and one that can commit to changes for the product being developed (Ann et al., 2010).

Agile allows the developers to provide an incremental total cost estimate at a detailed level as the iterations are performed. The big challenge with moving to an incremental costing approach is that the contracting cycle takes too long for just one development iteration.

According to Jim's research (2014) found the main problems with agile. Its process doesn't begin with an understanding the user requirement, it does not include both interaction designers and any user evaluation. In the original conception of agile development, there was no user experience design role and that's why the user interface produces a poor usability.

The research question was extended further (Cooper, 2006) towards agile ideas. He presented a set of questions that need to be considered to introduce agility into the process from the requirement of software development process and applicability of agile principles for component selection.

Agile ideas benefit the requirements engineering activity for software mainly by their adoption in the processes of requirements elicitation and component selection. An

iterative selection process supports a fertile dialog with the customer and makes the requirements elicitation and selection process manageable and observable.

Alleman (2002) described the principles for managing software project in agile.

These principles are:

(i) **Assume simplicity**: as the project evolves, it is assumed that the simplest solution is best.

(ii) **Embrace change**: when requirements evolve every time, the users may change their point of view in development process of project.

(iii) **Enabling the next effort**: the project can still be considered as failure even during the team derivers the system to the users. The system is robust enough to be extended over time.The next iteration may release the new system or support the existing one.

(iv) **Incremental change:** project development process can change the system into the small project and remove the useless requirement in incremental manner.

(v) **Maximize stakeholder value:** the stakeholders are investing resources to create a system to meet their user requirements.

(vi) **Manage with purpose:** by creatingartifacts that have stakeholders value and identify who needs the artifact.

(vii) **Multiple project views:** there is need for a wide range of presentation formats in order to effectively communicate with the stakeholders, participants and service providers.

(viii) **Rapid feedback:** the time of the feedback must be minimized. Work closely with stakeholders to understand the requirements, to analyze those requirements and develop work plan which provides opportunities for feedback.

(ix) **Working software is the primary goal:** Any activity that does not directly contribute to the goal of producing working software should be examined to determine its value.

The highest priority is to satisfy the customer through the early and continuous delivery of valuable software,to allow changing requirements, even later in development. Agile processes harness change for the customer's competitive advantage. The agile team delivered software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. Business people and developers must work together daily throughout the project, give them the environment and support their needs, and trust them to get the job done. The most efficient and effective method of conveying information to and within a development team is face to face conversation.

2.3.1 Agile Methods

The agile methods allow for software development teams to focus on the software rather than the design and documentation (Sommerville, 2011).

Agile is really a development approach, and it comprises a number of more specific methods such as:eXtreme Programming (XP), Scrum, Crystal Methodologies, Dynamic Software Development Method, Feature-driven development, and lean software development (Ann et al., 2010). Extreme programming (XP) is probably one of the best known and most widely used agile methods (Jefferies et al., 2000).

Extreme programming was originally designed to address the needs of the software development by small teams who faced changing requirements and system environments.

XP reflected the four following principles: (1) Incremental development is supported through small, frequent releases (2) Customer involvement is integral and supported throughout the process (3) People are focus of the process, not the development process (4) Change is embraced as prototypes were constantly released to the user.

Traditional agile methods do include iteration and involvement of the business owners and user representative, but they did not include usability testing with actual users (Jim, 2014).

Scrum technique is an incremental and iterative agile software development framework for management of project, product or application development.

(Ahina et al, 2014) Scrum ‘s main principle is its perception ability that as soon as the product increments start developing, the user gets the better understanding of the system what they actually want (requirements). Scrum focuses on delivering the products quickly by maximizing the teams ‘ability keeping in mind that problems cannot be addressed understand fully.

Dynamic Software Development Method divides project into three phases (Kuda et al, 2011):

i. **Pre-project**

   The candidate projects are identified, project funding is realized and project commitment is ensured.

ii. **Project life-cycle**

   It has 5 stages that a project will have to go through to create an information system. The Feasibility Research and Business Research are sequential phases that complement to each other. After these phases have been concluded, the system is developed iteratively and incrementally in the Functional Model Iteration, Design & Build Iteration and Implementation stages.

iii. **Post project**

   The post-project phase ensures that the system operates effectively and efficiently. This is realized by maintenance, enhancements and fixes.Feature driven development combines model-driven and agile development with emphasis on iterative design.

Lean software development consists of six principles:

(i) Eliminate waste.
(ii) Amplify learning.
(iii) Decide as late as possible.
(iv) Deliver as fast as possible.
(v) Empower the team.
(vi) Build integrity

The research discussed on software development principles listed above by introducing agile practices that seek to eliminate waste, but the first issue is to learn how to identifythe waste by eliminating the extra features, extra processes, and waiting for things to happen.For illustration,

feedbackfrom end users adds complexity to a system, but it also adds considerable value as amplify learning among the team; per example for increasing feedback is the single most effective way to deal with complex software project.The development team should know their immediate customer and have ways for that customer to provide feedback. Create an environment in which capable workers canactively participate in running and improving their own work areas by giving the team access to the customers, and the team makes its own commitments might be needed.

Agile processes create options that allow decisions to be delayed until the customer needs are more clearly understood and the evolving technologies have had time to mature.

Measurements are important for tracking the progress of software development as a tool to see the whole processes,people will increase their performance in areas that are measured and Measurements should motivate the team to collaborate and find better ways to do things.

2.4 Human Centered Design

Human-centered design is a method of developing interactive systems that place emphasize on an ease way to use from the user's point of view. The human-centered design process was established in 1999 as ISO 13407 in ISO13407 international standard, where policies are prescribed regarding the development of systems offering high usability (Takeshi, & Shin'ichi, 2008).

The HCD process begins by examining the needs, dreams, and behaviors of people who wantedto benefit from its final product. The HCD team seeks to listen and understand what end usersprefer.

Human-centered approach helps to make sure that user needs are elicited and transferred in requirements, integrated, and evaluated throughout the product life cycle.

The basic principles of human-centered design as specified by ISO 13407 are: (1) To actively seek user involvement in the design process, and to fully understand the user and his or her task; (2) To appropriate allocate tasks between the jobs done by the user and the functions handled by the system; (3) To perform repetitive design and evaluation; and (4) to incorporate users of various type of skill and perspectives into the design process. According to (Giacomin, 2012) Ergonomics of human-centered system interaction which describes human centered design as an approach to systems design and development that aims to make interactive systems more usable by focusing on the use of the system and applying human factors/ergonomics and usability knowledge and techniques.

Giacomin (2012) also specifically recommends six characteristics for human centered design approach:

(i) The adoption of multidisciplinary skills and perspectives

(ii) Explicit understanding of users, tasks and environments

(iii) User-centered evaluation driven/refined design

(iv) Consideration of the whole user experience

(v) Involvement of users throughout design and development

(vi) Iterative process.

2.4.1 User Centered Design

User-centered design (UCD) is an approach to user interface design and development that involves users throughout the application design and development process. It does not only focus on understanding the users of a product under development but also requires an understanding of the tasks that users will perform with the system and of the environment (organizational, social, and physical) in which they will use the system (Debbie et al, 2005).

User-centered design is a process focusing on usability throughout the entire development process and further throughout the system life. UCD involves designing with an understanding of the users and getting the feedback on the design at various points during an iterative design process (Jim, 2014).

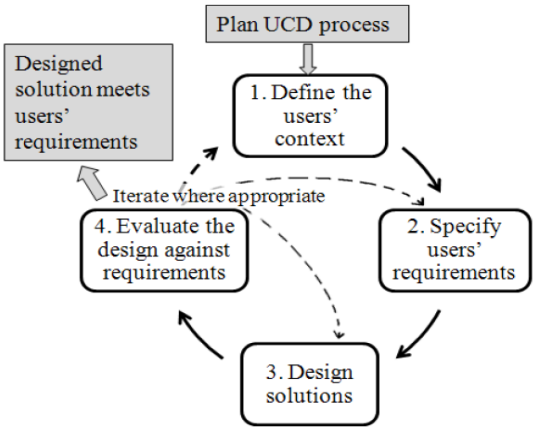

Figure 2.2: UCD process (Adopted from Schwartz, 2013).

Jan and Bengt (2003) describe that User centered design based on the following key principles:

(i) **User focus**: The goals of the activity, the work domain or context of use, the user's goal, tasks and needs should early guide the development.

(ii) **Active user involvement**: Representative users should actively participate, early and continuously thought the entire development process and the system lifecycle.

(iii) **Evolutionary systems development:** The systems development should be both iterative and incremental.

(iv) **Simple design representations:** The design must be represented in such ways that it can be easily understood by users and all other stakeholders.

(v) **Prototyping:** Early and continuously, prototypes should be used to visualize and evaluate ideas and design solutions in cooperation with the end users.

(vi) **Evaluate use in context:** Baseline usability goals and design criteria should control the development.

(vii) **Explicitly and conscious design activities:** The development process should contain dedicated design activities.

(viii) A professional attitude: The development process should be performed by effective multidisciplinary team.

(ix) Usability champion: Usability experts should be involved early and continuously throughout the development lifecycle.

(x) Holistic design: All aspects that influence the future use situation should be developed in parallel.

(xi) Processes customization: The user centered design process must be specified, adapted and implemented locally in each organization.

(xii) (xii)A user centered attitude: Should always be established.

The author combined the existing development process with industry standard user-centered design methods to create a detailed methodology that was compared to the usability of agile products.

2.4.2 User experience (UX)

User expereince is about how a software works on the outside,focuses on the users interaction with the system, user needs, user values, their abilities, and their limitations(Garrett, 2000).

User experience promote the improvement of interaction between users and perceptions of system and any related services.

Morville (2014) explained the quality of user expereience:

(i) Userful: thecontent should be original and fulfill a need, we must have the courage and creativity to ask whether our software are suseful.

(ii) Usable: the software must be easy to use.

(iii)Desirable: the software's design elements must be tempered by appreciation and emotional design.

(iv)Findable: the software needs to be navigable on website and locatable, the users must find what they need.

(v) Accessible:the software should be accessible to the users with disablities.

(vi) Credible: the users must understand and believe the design elements of the software.

(vii) Valuable: the software must deliver values to the stakeholders and improve customer satisfaction.

The main objective of Human Centered Design and User Experience is to understand people and their needs. Therefore, it is very useful to involve them directly to a project from the very beginning of it and consider their feedbacks in each iteration to reach the best solution. The research needsto try out solutions that work well for people and meet their needs.

2.4.3 Usability and User-Centered Design

Usability is viewed as an evaluation method and a design process, which is aimed at promoting learnability, efficiency of use, memorability, few and non-catastrophic errors, and subjective satisfaction of a targeted product (Preece, 1995). Software usability would need to take into consideration its information architecture, navigation design, iterative redesign cycle, user's personalized function, and heuristic evaluation (Pearrow, 2007).

Preece (1995) indicated that a user-centered design is a participant design in which users are invited to participate in the entire design process.

He formulated four design principles to engage users:

(i) Focus on users and their needs: users do not usually care about the organization is managed, they care how conducting the tasks and the goals achieved.

(ii) Focus on the functions to satisfy the needs after completing the analyses of user tasks and task environment.

(iii) Execute a formative evaluation in order to ensure that the system's functions meet the users' needs. Prototype can bring the desired solutions and by using ideation of think carefully to bring the solutions.

(iv) Use an iterative redesign cycle. It is important to find new solutions and options rather than choosing predetermined ones

Manuel and Antonio (2006) define usability in four sub-characteristics:

(i) Understandability: The capability of the component to enable the user to understand whether the component is suitable, and how it can be used and conditions of use.

The system designer should be able to design suitable element of interfaces for example menus and easy understand. The purpose of the system should be easily understandable.

(ii) Learnability: The capability of the software component to enable the user to learn its application. The user documentation and help should be consistent, help should be context sensitive and explain how to achieve common tasks, and the system should be easy to learn.

(iii) Operability: The capability of the software component to enable the user to operate and control it. The interface actions and elements should be consistent, error messages should explain how to recover from error, and the system should be customizable to meet specific user need.

(iv) Attractiveness: Thecapability of software component to be attractive to the user. The screen layout and colors should be appealing.

Lazar (2006) indicated that software usability is a matter of a user-centered design approach. He emphasized that a software must be designed with the user in mind in order to maximize the user experiences. The user-centered development life cycle is to ensure that the user needs should be the central issue of the software's design, from its conception to its implementation and management. Collecting requirements, designing

software's pages, and performing usability testing, the same researcher indicated, the designers keep in mind this focus, which would bring about a more effective design and more satisfied users.

In addition, (McLoughlin & Luca, 2002) hold that a user centered approach could help to develop learners' team skills. They emphasized the importance of understanding users in order to build better products. (Pearrow 2007) also confirms that a user-centered design is the central issue of software usability, which is based upon the user information.

2.5 Combining Agile and Human Centered Design

Human computer interaction (HCI) and software engineering (SE) are recognized as professions made up of very distinct populations. Each skill set is essential to produce the quality products, but no one set is sufficient on its own (Baxton, 2003).

As software engineers are also responsible for implementing User Interface Design (UID) specifications as running code, there is a need to communicate with HCI personnel. The interaction layer as interface between system and user is the area where HCI and SE are required to work together, to ensure that the resulting software product behaves as specified in the initial requirements engineering.

Faced with the demands of today's project environments, a reliance on heavy-weight methods such as formal models (SE), or massive documentation such as style guides (HCI) are far too expensive for the design of interactive products in any case. After all, greater cost-effectiveness and flexibility were among the main arguments for lighter, agile methods. In contrast, cheap and light-weight methods such as paper prototypes (HCI) or essential use cases (SE) are too abstract to be understood by everybody and too unstructured to guide RE towards a system specification. As agile approaches already exist in both SE and HCI (Gundelsweiler et al., 2004), they can be the interface for a common and balanced software lifecycle.

Nodder and Nielsen (2009) added that after examination there are three different approaches that can be taken for combining human centered design and agile:

(1)Human Centered Design (HCD)practices are added in to agile development framework. This allows organizations that are already doing agile development to deliver more usable system.

(2) Agile practices are added to HCD framework. This approach aims to make the HCD process more agile, not necessarily to create a comprehensive software development framework.

(3) Balance integration between agile development and human centered design.

This research has adopted the balance integration between agile and human centered design to find tangible issues that can be addressed to facilitate agile developers and UCD practitioners to work together and deliver the quality software with user experiences.

Nodder and Nielsen (2009) described that the integrated life cycle work within integrated systems development process can be divided into four group:

1. Early work for example site visits and another user research that happens before the actual implementation start.
2. Sprint specific work example detailed UI design and usability evaluations.
3. Decoupled work or work that happens outside sprint example additional research or stakeholder's workshop.
4. Post-print work example usability validation or UI redesign and refactoring.

Besides, the modification of requirements according to agile software available on the market may results in the change of business strategies, which become a great risk. In the selection of agilesystem, the requirement is the first specified and prioritized.

However instead of developing system, the activities focus on matching the requirements to the final product.

2.6 Summary and Research Gaps

The area of research that has not been widely studied in literature review is the agile usability testing, while usability problems may not lead directly to failure. The agile product development doesn't provide the entire solution for user satisfaction in terms of user experience. Users can interfere with productivity, making harder for users to achieve their goals as effectively or efficiently as possible.

ANodder and Nielsen (2009) suggested in the literature, that the research need to improve approach for adding agile practice in human centered design framework by focusing also on user interfaces in the development process. This research was focused on the software products because of their widespread usage. The contention of this research was that improvement of agile development usability can be achieved by designing the system with all requirements needed by users and easy interactivity of the system. The applicability of the human centered design (HCD) and agile studied to achieve the usability of the system and all requirements needed by customers.

CHAPTER THREE

METHODOLOGY

3.1Introduction

This chapter described the research methodology, the design used in the study, including data collection and analysis, target population, proposed framework and testing tool used in this research.

3.2Research Design

This research used a testing web performance methodology to evaluate usability of software products using a testing tool known as “Loadcomplete”. This tool enabled the tester to evaluate the performance of web application. The performance testing determined or validated the speed, scalability, and/or stability characteristics of the system or application under test. The performance was concerned with achieving response times, throughout, and the resource-utilization levels that meet the performance objectives of the project or product. Performance testing represents the superset of all the other subcategories of performance-related testing (Neha et al, 2015). The design was very appropriate because due to its ability to elicit a diverse range of information about the area of study.

3.2.1 Samplingtechniques

The sampling of this research was purposive sampling. The purposive sampling is a type of non-probability sampling that is selected based on the knowledge of a population and the purpose of the study, the subjects are selected because of some specific characteristic the research is looking for, and depending on what is investigated (Levy &Lemeshow, 2008). The sample size was 10 virtual users that accounted from usability testing.

In user testing, the research focused on software's functionalities to see which system elements were easy or difficult to use. According to Jeff (2015), the best way to make a sample size is to (i) pick the minimum problem percentages a researcher want to detect (margin of error) (ii) identify how sure the researcher want to be, and he will see these issues in a usability evaluation (confidence) as showed in table 3.1 for sample size and (iii) use the binominal probability formula to determine the sample size.

$$S = \frac{\log(1 - D)}{\log(1 - O)} \quad (3.1)$$

D = Change of detecting

O = Probability of occurring

S = Sample size

Table 3.1 Sample size

	Sample Size Needed	
Margin of Error (+/-)	90% Confidence	95% Confidence
24%	10	13
20%	15	21
15%	28	39
14%	32	46
13%	38	53
12%	45	63
11%	54	76
10%	65	93
9%	81	115
8%	103	147
7%	136	193
6%	186	263
5%	268	381
4%	421	597
3%	749	1,064
2%	1,689	2,398

Table 3.1 shows that based on 10 to 13 virtual users greatly increase the expected level of problem discovery but based on 15 to 21 participants has far less impact. These numbers are very important to know how to sample based on Table 3.1. A researchmay also have fewer virtual users if there will be opportunities to find important problems during software performance testing, so these tests often require fewer participants. Jeff (2015) suggested that good baseline was between five and ten participants. In general, there should be more participants for more complex, highly critical projects; while fewer participants were necessary when testing user interface.

3.2.2 Data collection procedures

Data collection is the systematic approach to gather and measure information from a variety of sources to get a complete and accurate picture of an area of interest (Kumar, 2013).

The information of value minimum, maximum and mean times of page access, load, and response were gathered in loadcomplete tool.

3.2.3 Data analysis

The data analysis consisted of examining the survey for correctness and completeness, coding and keying data into a database and performing an analysis of descriptive responses (Kumar, 2013).

After collection of data, the datawas coded and entered into Microsoft Excel sheet software for analysis.Excel sheet was used to display results in frequency tables and descriptive statics.

3.3Conceptual Framework

The research presented the proposed conceptual framework in order to improve the effectiveness and usability of interactive in software project by combining Agile with the Human Centered Design approaches.

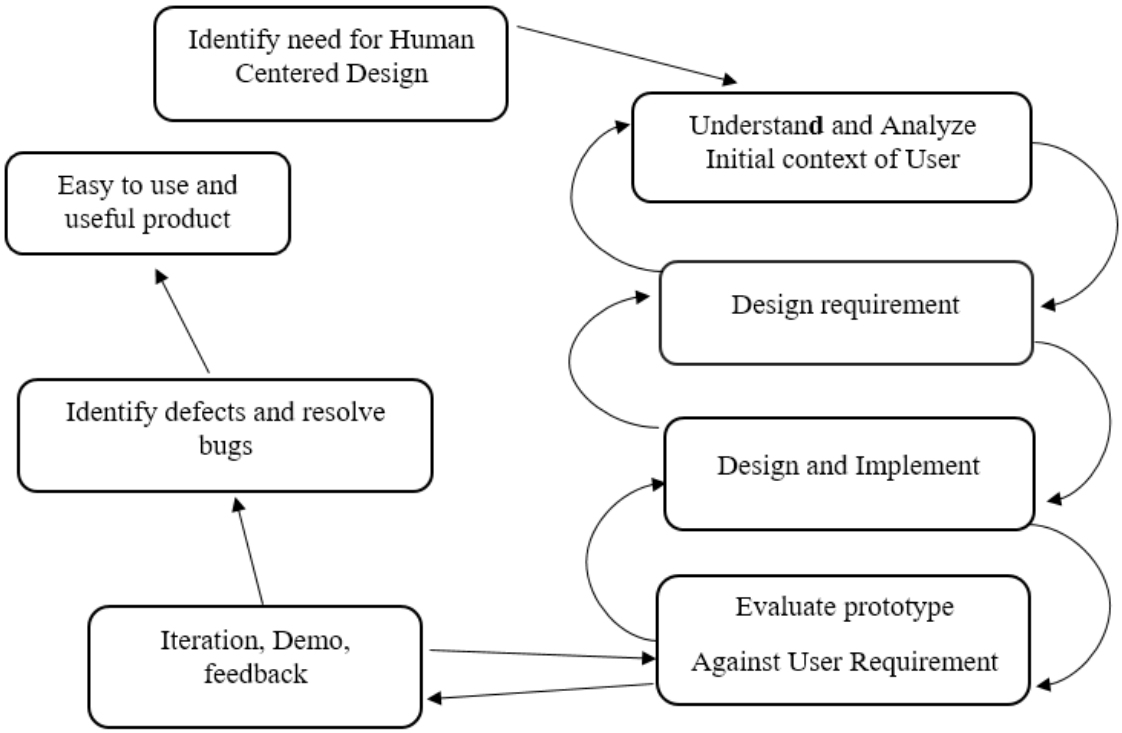

Figure 3.1: Conceptual framework

The conceptual framework is characterized by iterative cycles as shown on figure 3.1, and it has eight phases (iterations) including: identifying of the need for human centered design, understanding and analyzing the initial context of user etc. In iterative development, feature code is designed, developed and tested in repeated cycles. Improvement can quickly be recognized and implemented throughout each iteration, by allowing the next or previous iteration. Each iteration was explained in details in the following page.

3.3.1Identify need for Human Centered Design

Identify user need is the most important in software project, it's impossible to move forward in the right direction without considering users' requirement. The team should identifywhat the role of the users:Who they are, the roles that they play, what they want, what they like, what they know, what they think is useful, efficient and effective. The team should also identify what the users need to achieve as objectives.

3.3.2Understand and analyze initial context of user

This step helps to specify, in an organized way, the features of users. The team should take information from stakeholders, process it to full understand the business process and user goal in the context of the overall organization, the user's tasks and the business process that supports goals.

3.3.3Design requirement

This step state the essential characteristics that the solutions you get must meet to be successful. Based on the problem statement, a successful system would use less cost and delivered on time.

The best ways to describe the design requirement for the new system is to use the related, and existing product system. Investigate it in detail, take pictures and take it apart if you have authorization, analyze how and why it works the way it does.

3.3.4 Design and implement

This step helps to define and describe the necessary functions to achieve the objective of the system. The software allocation activities begin with the identified software functions obtained after completing the system allocation activities. The software functions are analyzed in term of feasibility.

3.3.5 Evaluate prototype against user requirement

The mockup, prototypes used as evaluations tools of functions requirement. The evaluation should be with users who have background knowledge and expectations for the system. The users should be asked to perform the representative tasks that the system has been designed to support.

The important for making test early in the development process is that possible to make modification or changes to the design without using high costs.

3.3.6 Iteration, demo and feedback

In development system with agile, the developer's team should fix the duration of the development process of the project. The concept of iteration helps developers to focus on features that are important to users and to test these features with a set of user tasks. Developers should work using iteration by selecting and implementing smaller tasks according to plan from the outstanding functions until they have done all tasks that have been assigned to them.

3.3.7 Identify defect and resolve bugs

After improving the functions of the system done during test, the team should identify the defect and bugs that escape during the iteration time.

The defect can have discovered after functionality that has been accepted by the product owner. The developer should fix any broken functionality as soon as it is found in iteration it is covered. Defects should have founded in quality assurance testing phase.

The quality assurance team should identify the defect and bugs when development system is done. All problem identified during quality assurance testing phase are created as defects because development is done.

3.3.8 Easy and useful product

Management of change clarifies the process by clearly describing to the users all required steps and prompting each user through his/her tasks. Management change helps stakeholders to improve visibility, accountability, prevent them from skipping steps and will reduce risks in the system environment. The useful product should have the user guide which, helps the users for the introduction of the system and how to learn to use it.

3.4Testing Tool

Software testing is the process of executing a program to verify its functionality and correctness (Thakur, 2015). The research used loadcomplete as the web performance testing tool for evaluating load, response, and availability of software products.

For creating tests for LoadComplete tool, the research started by recording HTTP requests and responses by consideringuser scenario. In its simplest form, the user would log into the web application under test by clicking some hyperlinks and the tool records the behavior of the system. A user scenario is then constructed from the observed HTTP frames being sent and received.

3.4.1 Approach followed

Odoo application and the prototype were tested using LoadCompletetool.

Odoo application was chosen as a product which is developed using agile approach and downloaded from the website (https://www.odoo.com/page/download).

Odoo is an OpenERP, everyone has the accesson the source code.There is no lisense fee associated with using Odoo.Odoo was chosen in this research because the configuration and installation are easier and quicker.Odoo is a fully web-based application. It can be accessedvia web browsers and it provides multi-user management support without any disturbance.

There are several different ERP software systems developed using agile methodology, for example, QuickBooks, Sage, and Sap. These softwaresare not easy for testing. A license is required for testing them while Odoo is easier for testing without a license.

The prototype model was developed using agile with human centered design techniques, and it was coded using PHP language in object oriented concept and the backend database was MySQL.

PHP is a popular general-purpose scripting language that is especially suited to web development. The PHP Hypertext Preprocessor (PHP) allows web developers to create dynamic content that interacts with databases. PHP is basically used for developing web-based software applications. PHP is a widely used, free, and efficient alternative to competitors such as Microsoft's ASP.

MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications.

Both Odoo and the prototype have the same features for sales, invoicing and reports. Sale has customers/clients, quotations, proforma, and invoice. In this research, the test was performed on sales features by comparing Odoo and the prototype. The researchtested two projects by recoding the interaction with odoo and the prototype web application. During the scenarios, loadcompletetool captured all HTTP and WebSocket traffic between the machine and the application server. The research recorded all performed scenarios for clients/customers, the recorded scenarios were simulated using the single virtual user and the number of virtual users was increased for three, five, seven and to ten virtual users. In this research, steady load was the type used for sending the accepted specified number of virtual users at once to the server. After the load test run had beencompleted, the loadCompletetool provided the results of simulated requests and received responses, presented and discussed in chapter four.

CHAPTER FOUR

RESEARCH RESULTS AND DISCUSSION

4.1 Introduction

In this chapter, results from performance tests are presented. The test was performed by analyzing the network traffic. Moreover, the page was loaded and the response time was considered. All the tests were executed on AMD A10 quad-core 2.20 GHz processor machine with 8GB ram, Microsoft Windows 10 with the internet speed of 7.2 Mbps. The tool used was loadComplete and the sample web applications were Odoo ERP and the prototype was developed during the research.

4.2 Test Results

The research used Steady load profile as test.This type allows sendingof the specified number of virtual users at once toa server.The test results included page load time, response time, memory, and CPU utilization of both Odoo ERP and the prototype.The interfacesin figure 4.1 and figure4.2 were tested and evaluated in this research.

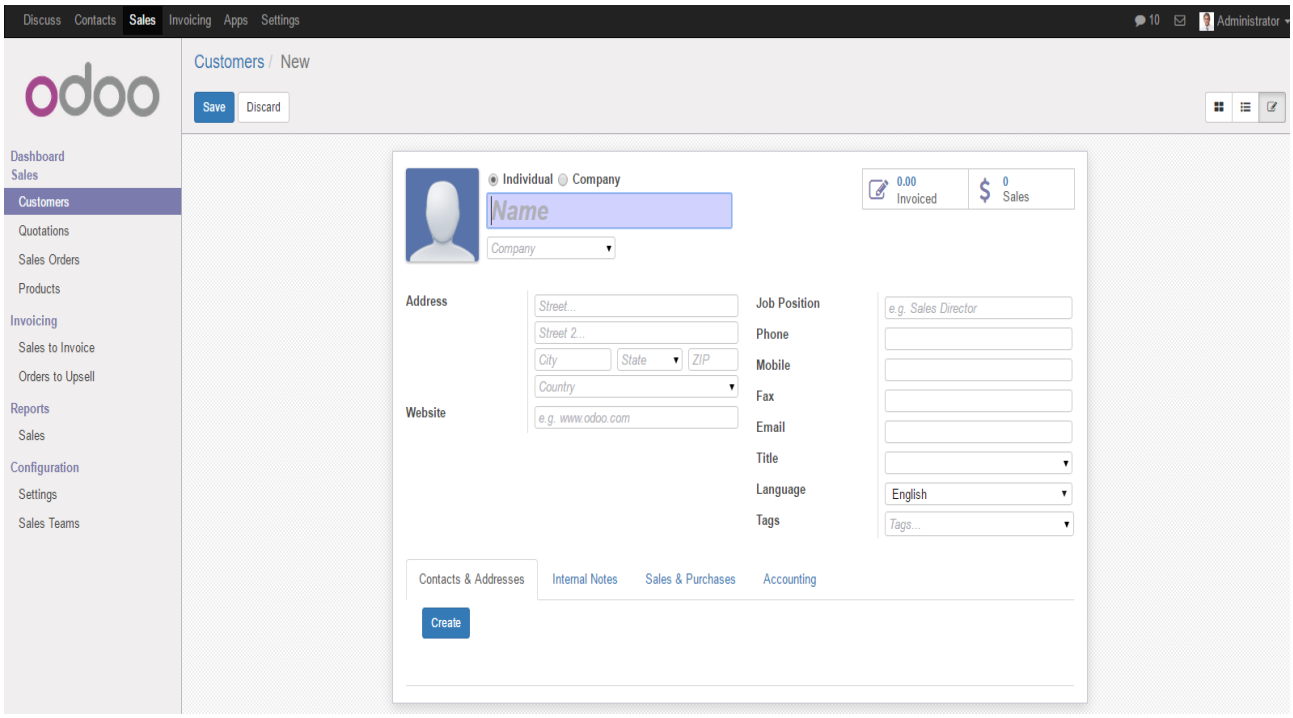

Figure 4.1: Odoo interface for client

Figure 4.1 representsOdoo ERP application which was taken for testing purpose.

This interface helped the user to fill all information related to customers in a database.

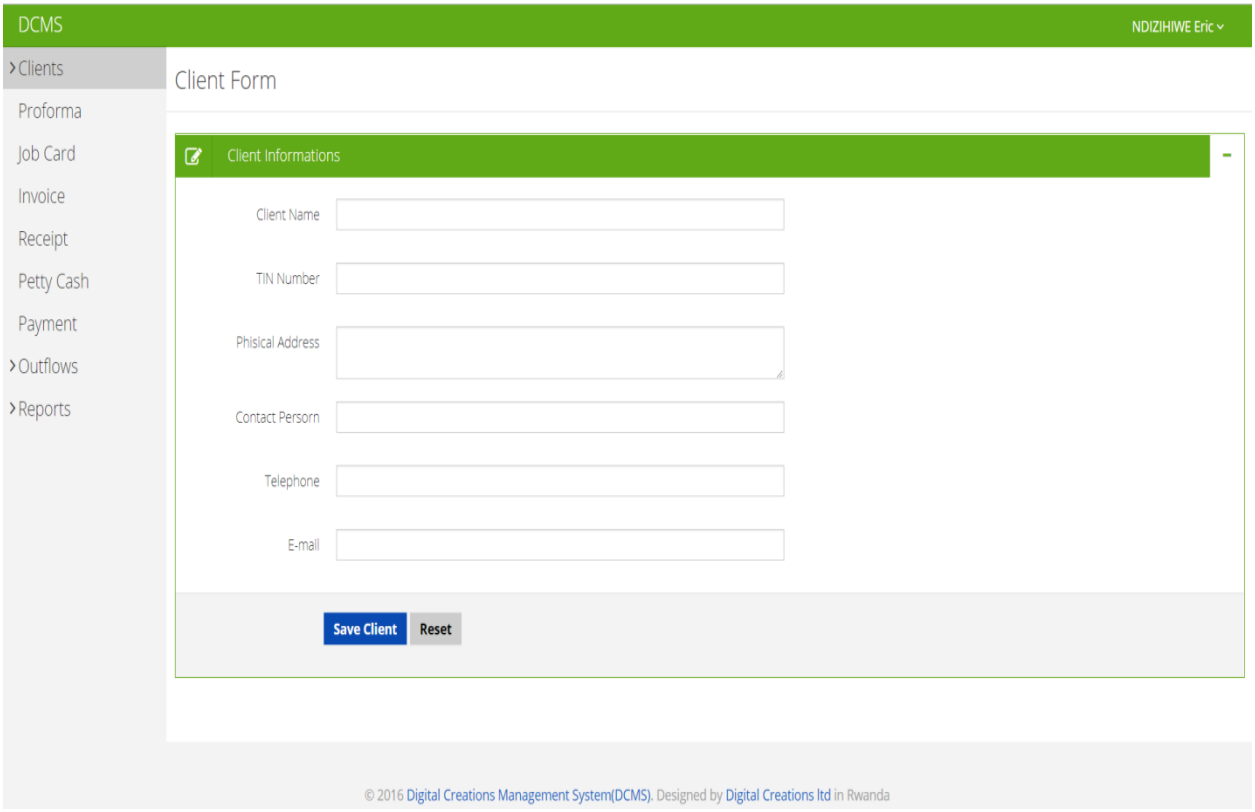

Figure 4.2: Prototype interface for client

Figure 4.2 representsthe prototype model application which was developed for testing purpose. This interface also helped the user to enter all information related to customers into a database.

4.2.1Page load time between Odoo ERP and Prototype

Page load time is the time that a page load takes to download and display the entire content of a web page inthe browser window.Page load time represented in Figure 4.3 shows Odoo ERP (in blue) and the prototype model (in orange).The x axis represents the virtual users for the system to load and the y axis represents the page load time in milliseconds .The load time was changed according to the number of virtual users.

The observation about theload page on Odoo and the prototypecan summurized as follow:

(i) **Load Page for Odoo ERP:**The load time of one virtual user took 22340 milliseconds, three virtual users took 22090 milliseconds, five virtual users took 21960 milliseconds, seven took 2190 milliseconds and ten virtual users took 24100 milliseconds.

(ii) **Load page for prototype model:**The load time shows that one virtual user took 7360 milliseconds, three virtual users took 8340 milliseconds, five virtual users took 8050 milliseconds, seven took 7340 and ten virtual users took 7770 milliseconds.

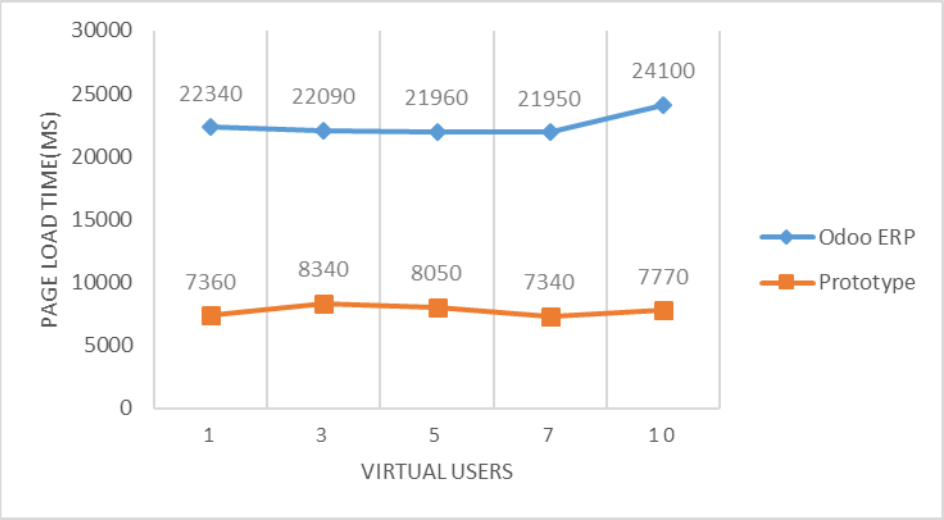

Figure 4.3: Pageload time between Odoo ERP and Prototype

The results retrieved from the test tool as shownin figure 4.3 indicates that the page load for Odoo ERP took more time to load than the prototype application. The average for page load time was 22.4 seconds for Odoo EPR developed in agile approachwhereas theprototype developed in agile with human centered design took 7.7 seconds.

This resultshowsthat the load page took much longer time to be loaded in Odoo applicationmore thanthe prototype model. Once the Odoo and the prototype model are compared, it is obvious that the Odoo ERP can decrease productivity withtaking long time for loadingwhich have always been the big problem experienced by users.

4.2.2Response Time between Odoo ERP and Prototype

Response time is the time spent by the server on responding to a request. The response time is important for your web application's success. The faster a web application communicates with a server the better it is for users and business.

The results for response time on Odoo and the prototype are the following:

(i) **Response time for Odoo ERP:**The response time of one virtual user took 50380 milliseconds, three virtual users took 51050 milliseconds, five virtual users took 53290 milliseconds, seven took 57870 and ten virtual users took 62030 milliseconds.

(ii) **Response time for the prototype model:** The result shows the time spent by the server for responding to a request: One virtual user took 2860 milliseconds, three virtual users took 4930 milliseconds, five virtual users took 4540 milliseconds, seven took 1270 and ten virtual users took 52400 milliseconds.

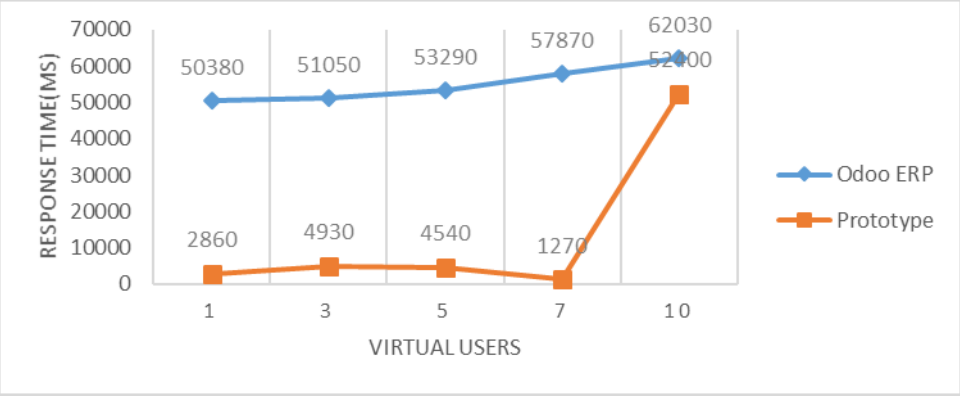

Figure 4.4: Response time between Odoo ERP and Prototype

The results obtained from Response time test shows that the prototype developed using agile with HCD gives better results compared toOdoo ERP developed using agile.

From the above results, it is observed that response time of odoo is greater than prototype response time as the number of virtual users is increased. Once the average of

response time for both systems,the prototype and Odoo ERP,the prototype had the response time average of 13.2 seconds whileOdoo's response time averagewas 54.9 seconds. The more response time decreases, the higher productivity is achieved. Generally, the rapid interactions are preferred. The end users have settled expectations based on their past experiences about the time required to complete their tasks using software, if the task is completed more quickly than expected the user will be appreciated.

4.2.3CPU Utilization for Odoo ERP

The percentage of Central Processing Unit (CPU) used throughout usage while running system is the main indicator of server strength and software system efficiency. Figure 4.5shows both the number of virtual users and the CPU usage statistics at any moment during the test period.

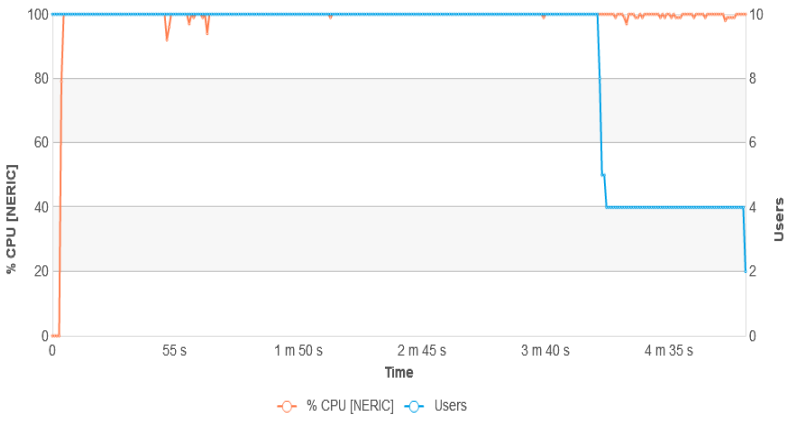

Figure 4.5: CPU Utilization for Odoo ERP

The light blue line showsthe virtual users from one to ten, while the dark orange line presents the CPU utilization. During the web performance tests, the timing was recorded. The interval time for the testgoes from 0 to 4 minutes and 35 seconds. Ten virtual users test showed that the average usage of CPU is 99%.As shown in figure 4.5, the CPU usage has high stress to the hardware.

4.2.4CPU Utilization for Prototype

Figure 4.6 represents both the number of virtual users and the CPU usage statistics during the test.

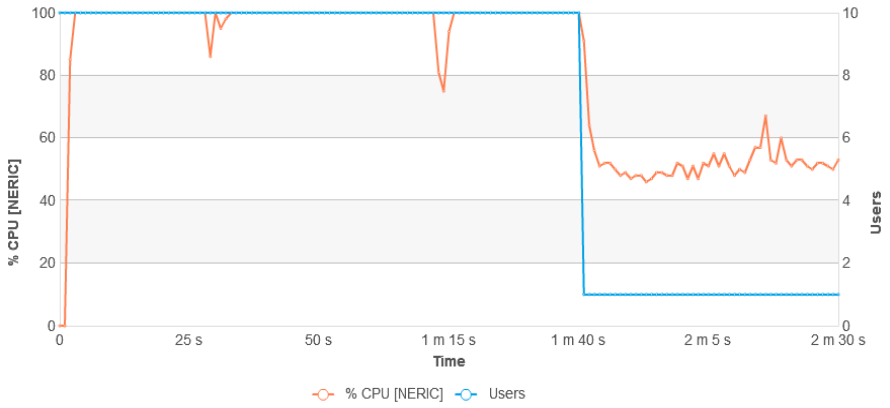

Figure 4.6: CPU Utilization for Prototype

The light blue line shows the virtual users from one to ten, while the dark orange line shows the CPU utilization. The time for the test took 2 minutes and 30 seconds and the CPU usage was reduced to 53% from 100% as shown on below graph.

Unlike the Odoo test which produced critically high CPU usage of 99%, the prototype results show that the CPU usagewas under 53%. If the CPU utilization is low, it means that an application is designed for good scaling and fast response time. This provides a better user experience. When a server has a high CPU usage on average, it means that an incoming request will be pending onqueueand this the sign of poor performance.

4.2.5 Memory Utilization for Odoo ERP

Test results showed that Odoo ERP used 68% of memory on scale time from 0 to 3 minutes and 45 seconds as shown on Figure 4.7.

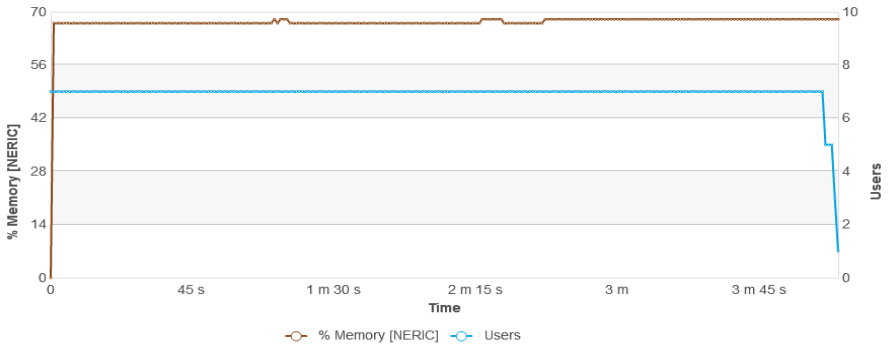

Figure 4.7: Memory Utilization for Odoo ERP

4.2.6 Memory Utilization for Porotype

The prototype used 67% of utilization memory in 1 minute and 40 seconds during the test as shown inFigure 4.8.

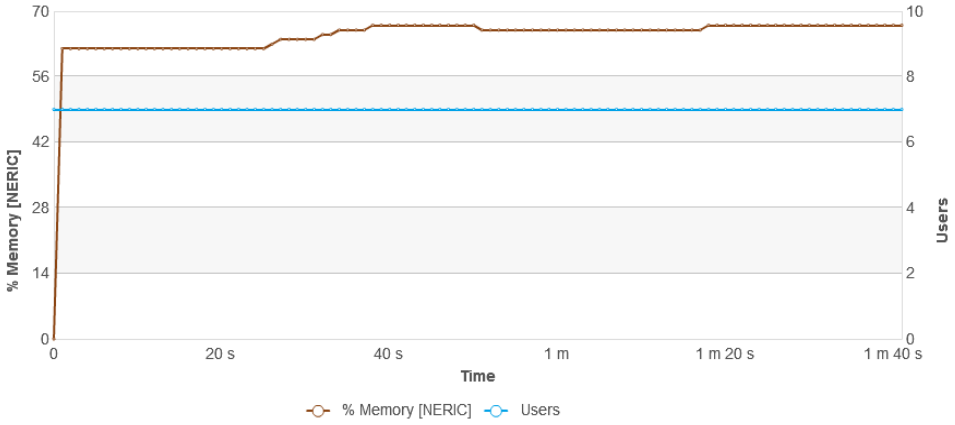

Figure 4.8: Memory Utilization for Prototype

The results from the test shows that both Odoo and the prototype consumed approximately the same memory usage (68% for Odoo and 67% for the prototype application). This managed the virtual user tests with comparably low stress to the hardware.

4.3

4.4 Discussion

The tests were performed using the four categories: page load time, response time, memory, and CPU as explained earlier. The research started by examining the load page between the prototype model and Odoo as shown in figure 4.3. When comparing the performance for both applications, it was clear that the load performance for Odoo is much loaded, compared to the prototype model which used less time.

An application which uses a load time of22.4 secondsdoes not scale very well. However, an application with a load time of 7.7 seconds scale much better. The result shows that prototype model counted 26% for total load page time compared with Odoo which counted 74%. The key issue is that if pages are not loading in an acceptable time then there's a risk that application indexing will run into problems. In fact, a fast load application helps to create an excellent and comfortable user experience.

The results for response time are shown in figure 4.2. Odoo application didn't respond fast more than the prototype. The performance increased from 54.9 seconds to 13.2 seconds. Response time has a significant impact on user experience. The result shows that response time took 19% for the prototype whileOdoo took 81%. A delay for application's response time can create an unpleasant user experience. Minimizing the average response time providesan output to the end users as quickly as possible and process their inputs as soon as they are received.

The observed results of the using CPU utilization are presented in Figure 4.6 for the prototype application and figure 4.5 for Odoo application, the researchnoticed that the less usage of the memory can improve the performance of an application compared to the highest memory usage. The CPU and memory usage cannot be too high, otherwise, the request processing can hang and system performance can be unstable. Futhermore, if the system resources (CPU and memory utilization) are too high, the system

performance may not be stable and user experience could degrade. When there are the high user requests on the application, always high CPU and high web server memory appear. During load test memory utilization did not affect the performance curve as shown in figure 4.7 for Odoo and Figure 4.8 for the prototype model, both Odoo and the prototype consumed approximately the same memory usage 68% and 67% for memory utilization.

CHAPTER FIVE

CONCLUSION AND FUTURE WORK

5.1Conclusion

The purpose of this research was to evaluate usability testing for software development; therefore theagile and human centered design(HCD) methodologies were chosen.When combined them in development process can increase usability for software. The prototypemodel was designed and developed using both metholologies.The research proposed and presented framework for combining agile and human centered design.

Odoo EPR as open source product developed in agile approach and the prototypemodelwere studied and tested. Various performance parameters like response time, load page and memory utilization were monitored. From this research, it was observed that the prototype model counted 26% for total load page time compared with Odoo which counted 74%, the results showed that agile software took time for loading page more than prototype. Response time took 19% for the prototype whileOdoo took 81%. By comparing Agile to Agile combined with HCD methodologies, the response time was high for agile software.

The aim of combined Agile and human centered design (HCD) was to reduce the risks from agile productsup to 100%. The researchshows that the high page loading time and high response time have the negative impact on usability of the system. The highest CPU usage was 99% foragile softwarewhile the CPU usage for the prototype was 68% during the tests. The developed model that combines agile and human centered design (HCD) for usability testing improved results for efficient and usability. The purpose of this research was to evaluate usability testing for software developmenttherefore,agile and human centered design(HCD) methodologies were chosen. Once both methodologies are combined in the development process, usability for software is increased. The prototype model was designed and developed using both methodologies.

The research proposed and presented the framework for combining agile and human centered design.

5.2 Future Work

Future work should look at the possibility of improving interaction and usability testing in software development by implementing interMod agile methodology with human centered design (HCD), The research has found that there is a lot of interest in software engineering area of research.

REFERENCES

Abrahamsson, P., Oza, N., & Siponen, M. T. (2010). Agile Software Development Methods: A Comparative Review1. In *Agile software development* (pp. 31-59). Springer Berlin Heidelberg.

Alleman, G. B. (2002, August). Agile project management methods for ERP: how to apply agile processes to complex COTS projects and live to tell about it. In *Conference on Extreme Programming and Agile Methods* (pp. 70-88). Springer, Berlin, Heidelberg.

Baxton, B. (2003). Performance by design: The role of design in software product development. *In Proceedings of the Second* I*nternational Conference on Usage-Centered Design*, Portsmouth, NH, ICSUC

Calisir, F., & Calisir, F. (2004). The relation of interface usability characteristics, perceived usefulness, and perceived ease of use to end-user satisfaction with enterprise resource planning (ERP) systems. *Computers in human behavior*, *20*(4), 505-515.

Cooper, K. M. (2006, September). Can Agility be Introduced into Requirements Engineering for COTS Component Based Development?. In *Software Product Management, 2006. IWSPM'06. International Workshop on* (pp. 35-37). IEEE.

Da Silva, T. S., Silveira, M. S., Melo, C. D. O., & Parzianello, L. C. (2013, July). Understanding the UX designer's role within agile teams. In *International Conference of Design, User Experience, and Usability* (pp. 599-609). Springer Berlin Heidelberg.

Easterbrook, S., Singer, J., Storey, M. A., & Damian, D. (2008). Selecting empirical methods for software engineering research. *Guide to advanced empirical software engineering*, 285-311.

Ferré, X., Juristo, N., Windl, H., & Constantine, L. (2001). Usability basics for software developers. *IEEE software*, *18*(1), 22-29.

Fowler, M. (2008). The new methodology. 2005. *Online at http://www. martinfowler. com/articles/newMethodology. html*

Garret, J.J. (2011). *The Elements of User Experience: User-Centered Design for the Web and Beyond*, (2nd ed.).New Riders: Pearson

Giacomin, J. (2012). What is Human Centered Design? *(Unpublished document). Human Centered Design Institut*e, Brunel University. UK

Gundelsweiler, F., Memmel, T., Reiterer, H. (2004). *Agile Usability Engineering*: New York:. NewSpringer,

Hocko, J. (2011). User-centered design in procured software implementations. *Journal of Usability Studies*, *6*(2), 60-74.

Jan, G., Bengt, G. (2003). *Key principles for user-centered system design*. Dept. for HCI/IT, Uppsala University. Retrieved 10/07/2015, from http://www.design4all.ch/data/KeyPrinciplesForUCSD.pdf

Jeff, R., Dana, C. (2008). *How to Plan, Design, and Conduct Effective Tests*. Wiley Publishing. Canada

Jeff, S. (2015). *How to Find the Sample Size for 8 Common Research Designs*. Retrieved, from http://www.measuringu.com/blog/sample-size-designs.php

Jefferies, R., Anderson, C. Hendrickson. (2000). Extreme programming installed. In: The XP Series. Longman : Addison-Wesley

Jim, R. (2014). *How user experience fits in Agile,* NJ 08512, Cranbury

Kazman, R. Bassl, L., & Bosch J. (2003) Workshop overviews: Bridging the gaps between software engineering and human-computer interaction, *In Proceedings of the 25th international conference on software engineering. IEEE Computers Society*.

Kuda, N.R., Kavita, N.G., Praneeth, C. (2011). A Research of the Agile Software Development Methods, Applicability and Implications in Industry. *IJSE l5(*2),.

Kumar, M. (2013). *Source of Data.* Retrieved 20/07/2016, from http://www.slideshare.net/manukumarkm/source-of-data-in-research.

Lapham, M. A., Williams, R., Hammons, C. B., Burton, D., & Schenker, A. R. (2010). Considerations for using agile in DoD acquisition. Australia. Australia. https://www.sei.cmu.edu/reports/10tn002.pdf

Lazar, J. (2006). *Web usability: A user-centered design approach*. Boston, MA: Addison Wesley.

Levy, P. S., &Lemeshow, S. (2008). *A sampling of populations: Methods and applications*. New York: Wiley & Sons.

Lucie, G. (2015). Human-Centered Design helps to create the best experiences. *Ecological Economics*, *56*(2), 280-293

Manuel, F.B.,Jose, M.T, Antonio, V. (2006). Measuring the usability of software components. *The journal of the systems and software 79*, 427-439.

McLoughlin, C., & Luca, J. (2002). A learner-centered approach to developing team skills through web based learning and assessment. *British Journal of Educational Technology, 33*(5), 571-582.

Morville,P. (2014). *User Experience Design*. Retrieved 14/07/2016, from http://semanticstudios.com/user_experience_design/.

Nodder, C. (2009). Agile usability: Report on best practices for user experience on agile development projects. Fremont, CA: Nielsen Norman Group. Preece, J. (1995). *A guide to usability: Human factors in computing*. Wokingham: Addison-Wesley.

Schwartz, L. (2013, October). Agile-User Experience Design: an Agile and User-Centered Process?. In *Proc. the 8th International Conference on Software Engineering Advances* (pp. 346-351). -.

Sommerville, I. (2011). *Software Engineering* (9th Ed.), Boston, MA. Addison Wesley

Takeshi, H., & Shin'ichi, F. (2008). Applying human-centered design process to SystemDirector Enterprise development methodology. *NEC Technical Journal*, *3*(2), 12-16.