A Usability Oriented Software Development Process with Constant Usability Evaluation

By
Muhammad Awais Gondal
2011-NUST-MS-PhD IT-004

Supervisor
Dr. Hafiz Farooq Ahmad
Department of Computing

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Information Technology (MS IT)

In
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(January 2014)

Approval

It is certified that the contents and form of the thesis entitled "**A Usability Oriented Software Development Process with Constant Usability Evaluation**" submitted by **Muhammad Awais Gondal** have been found satisfactory for the requirement of the degree.

Advisor: **Dr. Hafiz Farooq Ahmad**

Signature: _____

Date: _____

Committee Member 1: **Dr. Khalid Latif**

Signature: _____

Date: _____

Committee Member 2: **Dr. Ali Mustafa Qamar**

Signature: _____

Date: _____

Committee Member 3: **Ms. Sana Khalique**

Signature: _____

Date: _____

Dedication

I dedicate this thesis work to my parents who from the start encouraged, supported and indulged me. Also to my brother who is an unending source of motivation and guidance.

It is also dedicated to my friends and to my teachers with whom I have an exceptional and admirable relationship.

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Muhammad Awais Gondal**

Signature: _____

Acknowledgment

First and foremost praises be to Allah, the Almighty, on whom ultimately we depend for direction and guidance.

I wish to express my deep sense of gratitude to my supervisor Dr. Hafiz Farooq Ahmad for giving me a chance to work with him, in a research environment and polish my skills.

Words are inadequate in offering my thanks to the members of advisory committee for their encouragement and valuable input whenever required.

Finally, yet importantly, I would like to express my heartfelt thanks to my colleagues and fellow students who helped me directly or indirectly.

Table of Contents

List of Tables

List of Figures

Abstract

Usability has become an important aspect of software applications and it is gaining importance increasingly. A number of approaches have been presented to develop more usable applications. However, as their focus is mainly on usability tasks and not on software engineering principles, consequently software engineers have failed to adopt them. Moreover, usability evaluation which is very critical in improving usability, is also neglected by usability models. A complete software development process is proposed in this thesis that integrates usability tasks and evaluation methods with software engineering activities. It focuses on constant usability evaluation throughout the software life cycle process and helps take necessary action to screen out usability issues in earlier phases. The proposed process is very close to general software development process which will help software developers to adopt it and include usability aspects in software applications in a seamless way. The validity of the process has been assessed by developing different software applications by using the proposed process. The results show that the process helped in handling most of the usability issues and changes in requirements in early design phase. Moreover, better levels of usability and user satisfaction rate were achieved by using the process in contrast to the software projects that did not use the proposed process. Though, the proposed process involves additional usability tasks but there was no major impact in overall software development time as observed in different case studies. Additionally, it helped in saving significant time in the development phase with enhanced usability of the developed applications.

Chapter 1

Introduction

This chapter gives the basic idea of the concepts involved in this research. It also presents the background and motivation for this study. Moreover, it provides the hypothesis, gives an idea of expected results, and methodology to get and evaluate the results. Finally, it presents the structure of this thesis document.

1.1 Introduction

The concept of usability in software applications is not new and has its roots back in 1980's. The field got maturity in 1990's and in the last two decades it has gained adequate significance. Demand for more friendly, easy to learn and usable software applications, has increased.

In this thesis, we present a software development process that integrates usability tasks and evaluation methods in a general arrangement of software development activities. Usability tasks included in the process are close to general practices of software engineers that makes it an easily adoptable option. In the later sections of this thesis, we have discussed the details of our proposed method on the basis of its characteristics.

1.2 Motivation

A number of approaches, principles and methods have been presented to incorporate usability in software applications, and to evaluate usability of software applications. Most of these methods just give a high-level picture of activities that help in making software applications with better usability.

Some approaches try to present software processes for developing more usable applications.

However, usability has not been adopted as an essential part by software organizations and engineers. The major reason behind is that software engineers tend to follow defined software development processes that lack concepts of usability. Incorporating usability features in applications is not a simple task for software engineers. The approaches that consider usability as an essential part of software applications have their focus on usability activities only. Moreover, for many usability processes, usability is considered as a separate part which is totally excluded at times.

1.3 Hypothesis

The hypothesis for this work is mentioned as below:

- By adopting the proposed process, we can achieve higher levels of usability.
- The proposed process identifies changing requirements, and thus helps to deal with them.
- Usability issues are uncovered at very early stages of development, by applying the proposed process.
- Time required to perform additional usability activities is negligible.
- It is easy to adopt the proposed usability oriented process.

So, the null hypothesis for this work is as follows:

- By adopting the proposed process, we cannot achieve higher levels of usability.
- The proposed process does not identify any changing requirements.
- Usability issues are not uncovered, or uncovered at late stages of development.
- Time required to perform additional usability activities is significant.
- It is not easy to adopt the proposed usability oriented process.

1.4 Expected Results

It is aimed that through our proposed approach, usability issues will be uncovered at early stages of development. It will be easier for software engineers to adopt our process, than adopting other usability oriented approaches. In addition to this, better levels of usability can be achieved by using our proposed method as a software development process.

1.5 Methodology

To evaluate the hypothesis, different software projects were developed by using the proposed process. These projects were of different scope, genre and different sized teams developed them, so that reliable results can be gathered through this diverse pool of projects.

1.6 Structure

First, we discuss the concepts of usability and its various definitions. Than we present an account of different approaches and methods found in literature, which focus on achieving usability. We also present different methods that are used to evaluate usability of software applications. Afterwards, we present our method and provide details of its different phases. In the later sections, we present the results related to our proposed software process from case study projects and also specify future lines of research.

Chapter 2

Literature Survey

The field of software usability has various dimensions including defining usability, processes for achieving it, evaluating usability of an application, heuristics for software usability and many others. This chapter presents a survey on the published work related to basics of software usability, process models for achieving it and use of certain metrics for evaluating software usability. The survey aims to provide an insight of software usability dimensions and to discover different important perspectives of the topic. Moreover, it also aims to refine the research problem by analyzing different works that have already been done.

The chapter has been divided in four major parts; first it focuses on defining usability in the light of different studies and standards. Second, the literature survey presents process models for achieving software usability. Then, it presents an account of methods for evaluating software usability and tries to specify that what metrics are useful in evaluating usability of an application. And last, it presents some studies on usability in healthcare applications as evaluation of this research has been done majorly through healthcare IT case studies.

2.1 Defining Usability

The term software usability has been defined by a number of standards in different ways. As detailed definitions and concepts of usability are available in different studies and articles [7][8], this document focuses only on providing a precise introduction to software usability and the basic knowledge of usability concepts.

Usability has been defined by a number of ISO standards but the definition of ISO 9241-11 is widely used. The major factors of usability according to this standard are efficiency, effectiveness and satisfaction. Usability definition according to ISO 9241-11 is as follows:

> "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." [9]

- Effectiveness: the degree to which the intended goals of user are achieved.
- Efficiency: the relation of performance and the resources that have to be used to achieve the intended goals.
- Satisfaction: the extent to which the user finds the use of the product acceptable.

It should be noted here, that according to above mentioned definition, usability also depends on the type of users, their specific goals and the particular context in which the product is used. It can be deduced that usability is affected by the type of users (either expert or novice), what they aim to do with the product and the situation in which they are using the specific product.

According to Jakob Nielsen [10], usability is a quality attribute that depends on five components.

1. Learnability: How easy is it for users to accomplish basic tasks the first time they encounter the design?
2. Efficiency: Once users have learned the design, how quickly can they perform tasks?
3. Memorability: When users return to the design after a period of not using it, how easily can they reestablish proficiency?
4. Errors: How many errors do users make, how severe are these errors, and how easily can they recover from the errors?
5. Satisfaction: How pleasant is it to use the design?

Like ISO, IEEE standards are also considered significant in a number of fields including Software Engineering. IEEE standard 1061 focuses on Software Quality Metrics and defines usability as given under:

> "Usability is the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component." [11]

We can find many other definitions and explanations of usability in literature, which differ and relate with one another in certain ways. However, almost all of them consider usability as a quality attribute of a software application. Other characteristics and components of usability found in literature are utility, safety, accessibility, simplicity and some others. Components of usability are somehow interrelated to each other and also they should be considered according to the nature and domain of the software being developed.

2.2 Process Models for Software Usability

Various process models and approaches for achieving usability can be found in literature. This chapter presents a number of such approaches and processes, and intends to act as a single resource for getting information regarding them. Moreover, it also aims to present a detailed review of existing models for attaining usability, so that we can present a suitable usability aimed development model relevant to the needs of software engineers.

2.2.1 User-Centered Design

User-Centered Design states that user must be the focus of UI design, as the basic purpose of a system is to serve the user. While developing a user interface focus should be on the user and system design should also focus on UI requirements.

User-Centered Design focuses on the user in every stage of system development. The user is focused while planning the project, analyzing requirements and tasks of the user, understanding usability requirements and user needs, designing prototypes and getting feedback from the user, planning the project according to suggestions and feedback, developing and deploying the system, taking input from the user after deployment and making changes, thus, involving the user in every iteration of development. So, it can be said that user is involved throughout the process and user's needs and feedback are given most importance.

Usability design in UCD contains three stages; conceptual design, interaction design and detailed design. Conceptual design serves as a high-level representation for the whole system; interaction design focuses on specifying

the details regarding navigation and user interactions while detailed design digs deep into individual parts of interface.

User-Centered Design basically has its foundation on twelve principles [12] which are as follows:

- User focus
- Active user involvement
- Evolutionary systems development
- Simple design representation
- Prototyping
- Evaluate use in context
- Explicit and conscious design activities
- Professional attitude
- Usability champion
- Holistic design
- Process customization
- User-centered attitude

User-Centered Design helps designers of the software to understand user requirements in a better way, thus enabling them to develop a better software. Moreover, user's self involvement helps in building real expectations and leading towards higher level of satisfaction as the software developed is according to the user's suggestions and demands. Constant involvement also helps the user to learn the software application in a very less time. However, this process also has some limitations like if the end-user is ambiguous or not accessible or not willing to get involved in the process, then user-centered design cannot be used. Moreover, issues regarding communication and interaction with the user can also lead to the failure of this process model. UCD remains apart from software engineering tasks which is one of the reasons that UCD is not frequently followed by software development individuals and organizations.

2.2.2 The Usability Engineering LifeCycle

The Usability Engineering LifeCycle (UEL), presented by Deborah Mayhew [1], is an iterative approach that focuses on usability tasks for developing systems. This approach tries to relate software engineering (SE) activities with usability engineering (UE) activities and attempts to integrate UE activities with SE processes. The general phases of SE in the Usability Engineering LifeCycle are requirements analysis, design, testing, development and installation. Each phase consists of some usability engineering activities to ensure that a certain level of usability is achieved.

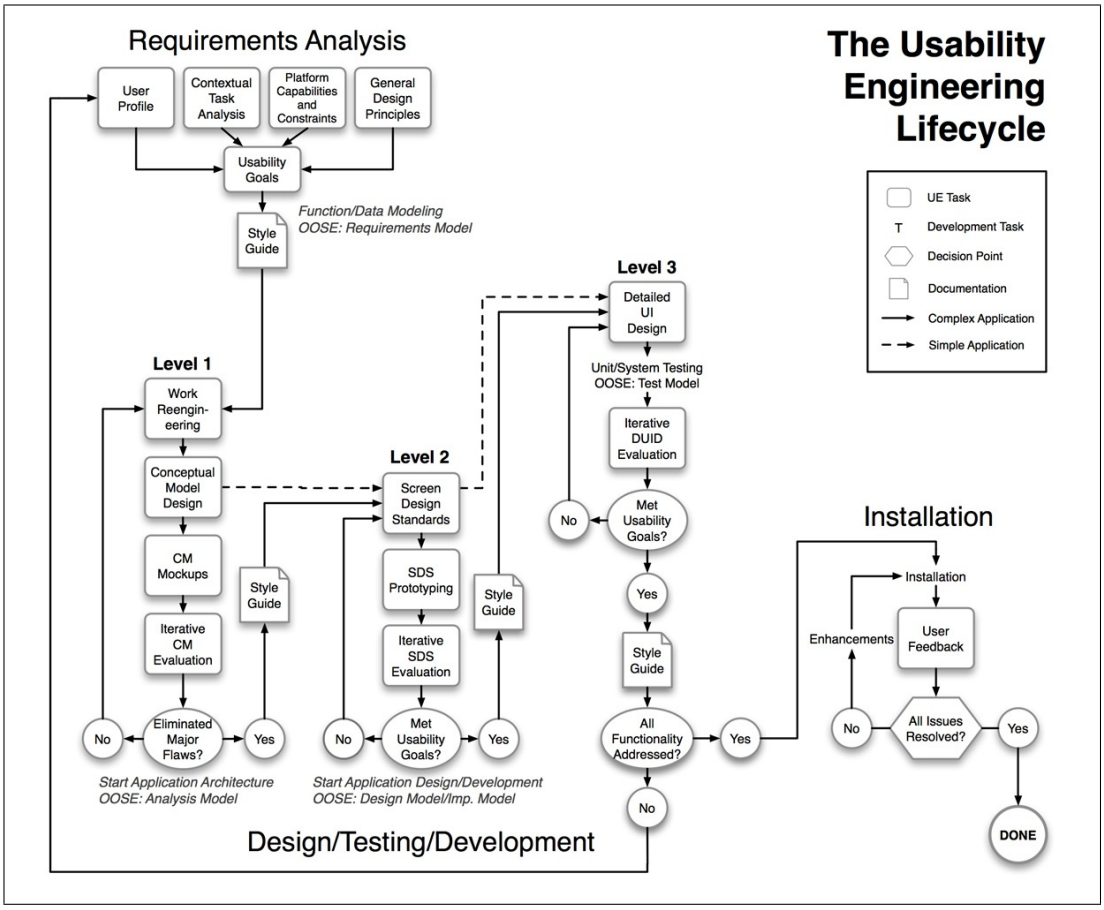


Figure 2.1: The Usability Engineering LifeCycle [1]

The requirements analysis phase involves the following activities:

- Creating the user profile with information related to expected users.

- Understanding and determining tasks, workflows and goals of the intended users.
- Specifying the capabilities and constraints of platform which can affect the design.
- Collecting available guidelines for UI as well as for design in that particular context.
- Formulating qualitative and quantitative usability goals through the gathered information (about users, tasks, goals, guidelines etc).

Next phase combines three general phases of software engineering i.e. design, testing and development. This phase also focuses on a number of usability engineering activities divided in three levels. Brief detail of activities in these three levels is provided below:

- Level 1: This level is concerned with re-designing tasks on the basis of organizational workflows, making high level design rules for defining navigational pathways, preparing paper-based mockups or prototypes and evaluating them with users.
- Level 2: It defines standards for all screen designs and applying them for designing working/running prototypes. It also involves evaluation of the prototypes and making style guide accordingly.
- Level 3: This level involves developing the final product according to the style guides, and evaluating the product.

The last phase of UEL is the installation phase which focuses on getting feedback from actual users after the product has been deployed, so that enhancements and betterments can be made in the UI design.

The Usability Engineering LifeCycle uses general software engineering phases and blends usability engineering activities with them to design usable systems. It advocates revisiting of certain levels if goals of that particular level are not met. It might help in achieving better levels of usability, but at the same time it can also increase development cost and time. UEL also supports improving the system through user feedback once it has been installed.

Although, UEL offers a base for usability researchers to present better usability processes, however, it does not provide much ease for software developers to merge usability and software engineering tasks. Not only software

engineering activities are missing in the process but also developers have to return to requirements phase if all functionalities are not addressed after development.

2.2.3 Usability Design Process

Usability Design Process (UDP) [2] follows the twelve principles of User-Centered Design and adopts the structure of Usability Engineering LifeCycle which has three main phases. UCD has its main stages as requirements analysis, growing software with iterative design and deployment. We present these three phases as follows:

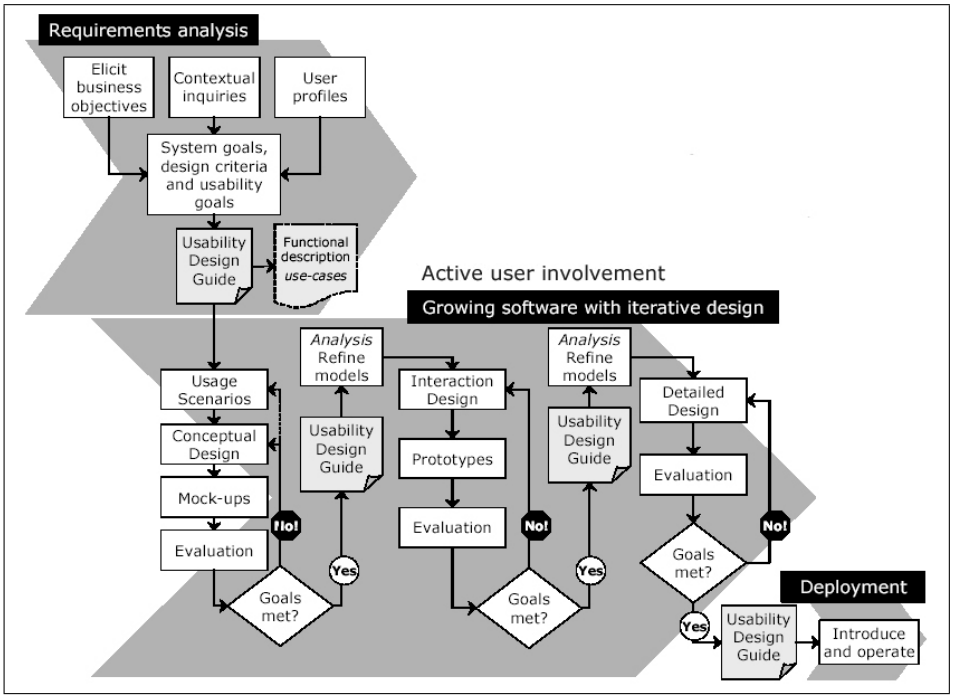


Figure 2.2: Usability Design Process [2]

Requirements Analysis focuses on elicitation of business goals and documenting them. It also focuses on gathering complete knowledge regarding use of the system and needs of users. Users are analyzed on the basis of their skills, knowledge and abilities. UCD also supports the use of contextual inquiries to have a complete understanding of what users do at work in actual and what are their goals, tasks and environment. Although, UDP advocates the importance of identifying system goals, design criteria and usability goals, yet it admits that discovering usability goals at initial level is

not an easy task. The information regarding business goals, user profiles, design criteria, usability goals etc helps in developing a usability design guide, which is specific to the project being developed. Usability Design Guide is separately discussed at the end of this section.

In growing software phase the system is designed and grown in three iterative loops as mentioned by User-Centered Design i.e. conceptual design, interaction design and detailed design. Conceptual design focuses on usage of sketches and paper based mockups to symbolize the system. Details regarding user interactions, screen navigations etc are specified in interaction design. Detailed design focuses on details of the interface like data fields, inputs, menus etc.

UDP considers deployment phase critical for the success of the project and promotes the planning of this phase from start. UDP supports that deploying the system incrementally helps in reducing resources for the process of deployment, and also it helps in making the system better with each increment.

As mentioned earlier, Usability Design Guide contains overview of the project, a plan for participation of the user, user profiles, task analysis, scenarios, detailed design (which mainly focuses on screen design), design artifacts and data regarding feedback and evaluation. The Usability Design Guide can be adapted for other projects as well with required modifications. UCD also supports dividing Usability Design Guide in various documents to be used in various projects.

It has been admitted in [2], that UPD is not a complete software development process and it must be framed in a more comprehensive process. An attempt to integrate UDP in Rational Unified Process has also been made.

2.2.4 ISO 13407: Human-centred design processes for interactive systems

ISO 13407, Human-centred design processes for interactive systems (now revised by ISO 9241-210) [13] presents an iterative development cycle for designing interactive systems. It is based on a repetitive process where context of use and user requirements are specified first. Then, a number of possible design solutions are proposed which are then evaluated according to user requirements. The process is repeated until all requirements are fulfilled. The

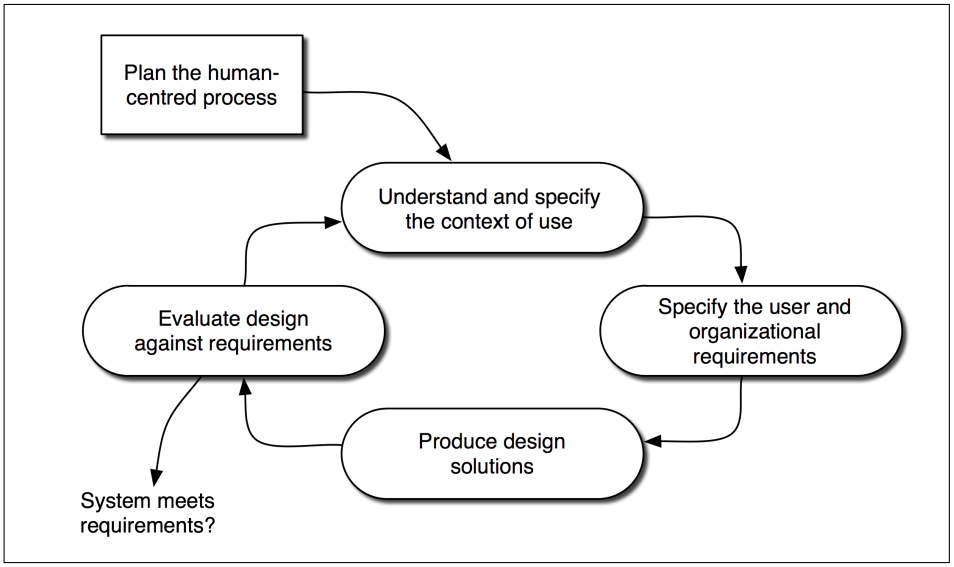


Figure 2.3: Human-centred design processes for interactive systems (ISO 13407)

cycle is a very general representation and it cannot be considered alone as a software development process, however, it can be adopted as a baseline with other process models.

The KESSU Usability Design Process Model [3] is an extension to the Human-centred design processes for interactive systems [13]. This model tries to explicate ISO 13407 model by specifying outcomes, best practices and sample methods. Figure 2.4 shows the visualization of KESSU model.

2.3 Evaluating Software Usability

In the previous chapter we discussed some process models and techniques that can be followed to develop a usable software. Now we present evaluation techniques and metrics that can be used to determine the extent to which a software is usable. The evaluation techniques have been studied in the literature [14] [15] and the metrics presented here have been extracted from published works and renowned resources for software usability testing and evaluation.

Generally, three main types of usability evaluation methods are found in the literature. These types are testing, inspection and inquiry.

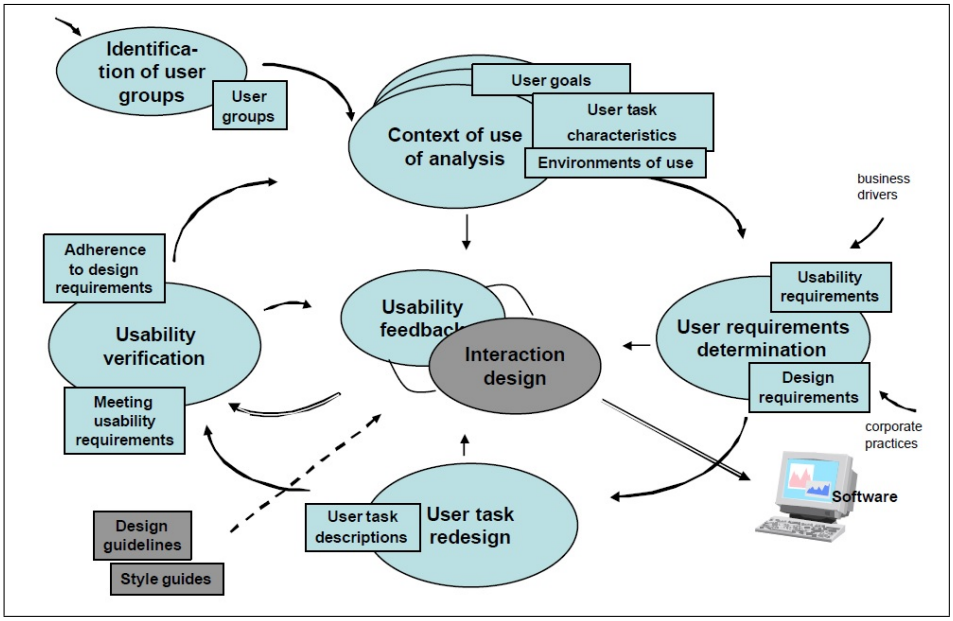


Figure 2.4: KESSU Usability Design Process Model [3]

In addition to this classification, analytic methods can also be considered for usability evaluation. In analytical methods, a model of the actual software product is used by representational users in order to test performance of the system [4].

2.3.1 Usability Inspection

In the inspection technique, professionals examine a system and evaluate the extent of its usability. The professionals can be trained usability experts, software developers or software quality engineers. In this section we discuss some of the major usability inspection methods.

Cognitive Walkthrough: In cognitive walkthrough technique, evaluators go through defined actions to complete specific tasks. They evaluate ease of use and understandability of the system on the basis of defined actions and their impact on effectiveness. Working prototypes can be used for this type of technique and it can easily be carried out in the design phase. This approach provides feedback regarding effectiveness of a system but it does not provide any quantitative data.

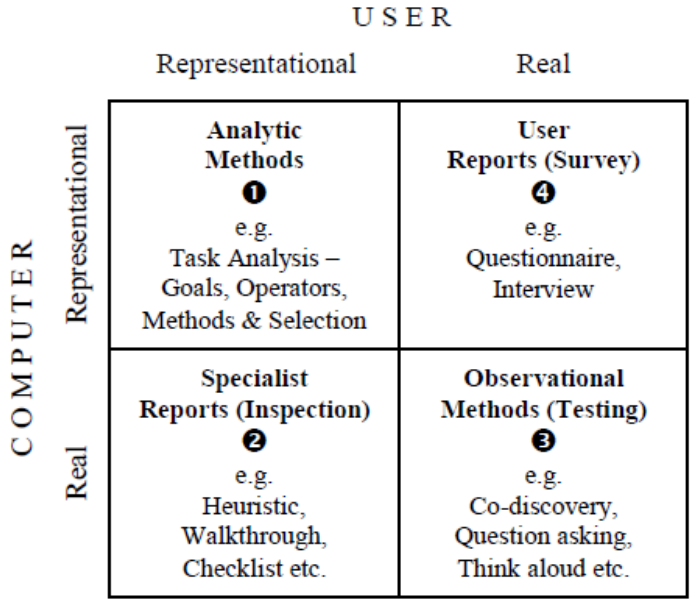


Figure 2.5: Classes of Usability Evaluation Methods [4]

Feature Inspection: This approach focuses on evaluating usability through inspecting features of a software product. The evaluators are given use cases of the software application on the basis of which they evaluate understandability, availability and other usability aspects. Moreover, it is also checked that either or not the features are properly organized and accessible without significant problems. Like cognitive walkthrough, this approach also does not provide quantitative assessment, however, they do judge the quality attributes of a product.

Heuristic evaluation: In heuristic evaluation, experts and usability testers evaluate a software product against a defined set of usability principles. Possible usability issues are uncovered as a result of the evaluation and some possible solutions can also be presented by the evaluator(s). Heuristic evaluation can be considered as the most common and widely used method for evaluating usability of software applications. Although many groups and individuals have presented heuristics and principles for usability evaluation, but the set of heuristics developed by Jakob Nielsen [16] are the most famous and are most used. As this set of heuristics is widely used and considered as most effective one, we summarize them as follows:

Jakob Nielsen's 10 Heuristics for User Interface Design: Nielsen and Rolf Molich developed the basic heuristics in 1991, which were later refined by Nielsen in 1994, after analyzing a large number of usability problems.

1. Visibility of system status: keep user informed about what system is doing.
2. Match between system and the real world: familiar language and real world terms should be used rather than computer terms.
3. User control and freedom: users should be able to undo/redo actions in case they have done something by mistake.
4. Consistency and standards: similar behaviors and terms should be used throughout the system.
5. Error prevention: error prone situations should be avoided or checked well to avoid any errors.
6. Recognition rather than recall: memory load of the user should be minimized.
7. Flexibility and efficiency of use: the system should be made flexible by giving users the option to tailor actions for any functionality.
8. Aesthetic and minimalist design: irrelevant information should not be displayed as it minimizes the space for relevant information.
9. Help users recognize, diagnose, and recover from errors: helpful messages should be shown which may help the user in recovering from a problem.
10. Help and documentation: documented help should be provided wherever required.

However, Jakob Nielsen's heuristics are not always equally effective and certain domains may require specific heuristics. A methodology for creating new heuristics for specific applications has been presented in [17] which presents defined steps to establish heuristics. Stages of specifying usability heuristics [17] are as follows:

1. Exploratory Stage: collect bibliography related to specific application.
2. Descriptive Stage: emphasize the most significant characteristics from bibliography.

3. Correlational Stage: discover the characteristics that should be there in heuristics for specific applications.

4. Explicative Stage: formally specify the proposed heuristics.

5. Validation Stage: validate the proposed heuristics through case studies etc.

6. Refinement Stage: refine according to feedback from validation stage.

Heuristics evaluation provides a quick feedback regarding usability problems and also helps in getting better solutions for specific problems. It can provide an early evaluation in the design process which leads towards a better and usable end product. It can also be conducted along with other usability evaluation techniques. However, heuristics evaluation should not be considered as the replacement of usability testing. It is recommended that heuristic evaluation should be done before applying other usability testing techniques, as this type of evaluation is better for identifying minor issues. Appropriate expertise and skills are also required for effective heuristics evaluation.

2.3.2 Usability Testing

In this approach, users use the system (or prototypes) according to defined tasks or procedures, and the evaluators assess the data and results from this specific usage. On the basis of the results, it is assessed that how much the system helps the users to accomplish their tasks. Major Usability Testing approaches are discussed as follows:

Coaching Method: In coaching method, users use a software product in presence of a coach (expert of the application or specific domain). Users ask questions regarding the usage of the product and the coach answers them accordingly. The basic purpose of this approach is to evaluate and find out the extent of guidance or help required for using a particular software application. A product will not be considered as usable enough if users want answers to a large number of questions, for using the product. Some extensions of this approach are also used in which answers are controlled and multiple answers are provided for similar type of questions. This helps in determining what kind of answers help users the most.

Performance Measurement: Performance Measurement can be considered as the most appropriate one, when quantitative data is required. It

can be really helpful in comparing usability aspects of different software applications and in evaluating against defined benchmarks. Quantitative data regarding application's capability as well as of participant's (user's) ability can be gathered. In this approach users use the product, (preferably in a usability lab environment,) and the testers gather certain data on the basis of the usage to draw conclusions. Certain goals are defined before the test is conducted and then quantifiable data is collected during the test. Examples of measurements taken during the test can be:

- Time taken to accomplish a task
- Total number of errors
- Time required for recovering from error situations
- The features that are used the most and the features that are not used

Although, performance measurement focuses on quantitative data but it is recommended that during the test, qualitative data should also be collected to get information regarding user's behavior, likeness and other aspects. Moreover, if users with different skill levels participate in the test, additional and better results can be concluded.

Question-Asking Protocol: In this approach, testers ask direct questions from the users while they are using the software application. Users are supposed to perform specified tasks and the questions asked might explore the behavior expected by the users or the level of difficulty for using the product. An example question can be, "How would you log out from the system?" The response might indicate the way expected by the users and in cases it can also tell if the users do not know how to perform this task.

2.3.3 Usability Inquiry

In this method, users are inquired about the usage experience regarding a system to get information about usage satisfaction, usability problems and requirements etc. Users may also be involved in answering a set of different questions related to usability of the system. Some usability inquiry methods are discussed in this section:

Field Observation: Testers observe users in real usage environment (workplace) and try to understand how users use the system to fulfill certain tasks.

The perception of users about the system can also be understood. A variation of this approach can also be used in early stages to get requirements for the system. Moreover, through observing the users it can also be determined that what usability problems they usually face. As most of the usability achieving processes are iterative, these problems may be fixed in later stages.

Questionnaires: In questionnaire method, a set of questions is designed regarding user satisfaction, product ratings, usability problems and others, and users fill the questionnaire to provide their opinion about usability of the product. This method might be considered as the easiest one and is most appreciated by the users. There are a number of questionnaires specifically designed for usability evaluation of a software application. Some of these are System Usability Scale (SUS), Questionnaire for User Interface Satisfaction (QUIS), Software Usability Measurement Inventory (SUMI) and USE Questionnaire. Standard questionnaires also provide defined scales for the gathered output(s).

Interviews: Like questionnaire methods, this method also focuses on designing questions for evaluating usability aspects of the product. However, in this approach users answer the questions verbally and an interactive session is carried out. This method helps in getting detailed information and interaction with the interviewer helps in getting input that cannot be gathered through questionnaire method.

Along with these discussed evaluation types, many efforts have been made to present usability evaluation models according to specific evaluation type and for applying in particular domain [18] [19]. Moreover, effectiveness and usefulness of specific evaluation techniques have also been studied and compared [20] [21] [22]. However, selecting usability evaluation methods primarily depends on the project being developed, the type of users and resources available.

An evaluation process for model-driven web development has been presented in [5] and an effort has been made to integrate usability evaluation at different stages of model-driven web development. The process specifies three different roles that are involved in evaluation at on specific stages. The roles are evaluation designer, evaluator and web developer. Evaluation designer establishes evaluation requirements, specifies and designs the evaluation. The evaluator executes the evaluation and creates opportunity for improvement in the form of usability reports. Finally, the web developer

analyzes the changes required and makes improvement reports.

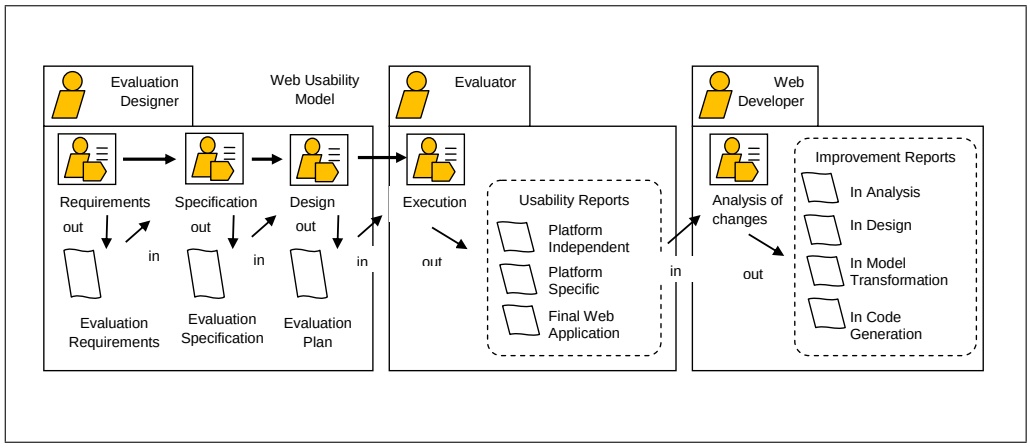


Figure 2.6: Web Usability Evaluation Process [5]

2.4 Usability in Healthcare Applications

Recently, usability has become a frequently discussed topic in healthcare information technology literature and standards. CCHIT (an organization that aims to efficiently accelerate the adoption and usage of healthcare systems through a convincing certification process), is also focusing on including usability as a criterion in the certification process. However, it is known from the literature that healthcare IT systems do have a number of usability problems that hinder prevalent adoption of HIT systems. Other aspects like change avoidance, organizational barriers, unavailable features and change resistance are also there, but usability has been acknowledged as a major issue that impede the adoption. The rate at which HIT systems have been adopted in hospitals is much less that the rate at which other fields have adopted IT technologies (Use of electronic health records in U.S. hospitals). It is required that effective process models and design principles should be there that help in developing highly usable HIT applications. In this section, we present the principles and models that help in achieving usability in healthcare applications or try to contribute towards it. We also present an account of known usability problems in healthcare information technology products.

As discussed in earlier chapter, multiple usability processes and principles can be found in literature, that help in developing highly usable software

applications. HIMSS EHR Usability Task Force has selected some principles [23] based on their impact on two major factors for healthcare application acceptance; efficiency of use and minimizing user errors. The selected principles have been briefly discussed here:

Simplicity: It implies that the design should not include information clutters and excessive information on the user interface. However, any required information should not be missed especially when it helps in decision making in case of healthcare applications. It is believed that this principle becomes more important when the application is complex. It is obvious that healthcare systems are complex with a sufficient amount of information required by its users, so this principle gets more magnitude when it comes to healthcare systems.

Naturalness: Naturalness submits that the application should be self learnable and must be close to the actual behaviors and procedures. Using real workflows and general practices may help in achieving naturalness. It must be noted here that healthcare workflows are very specific to consultant practices and must be reflected in software applications so that users can automatically learn.

Consistency: Consistency is an important principle for software applications in general and healthcare applications in particular. Consistent behaviors, layouts, icons, concepts and behaviors must be used so that users do not have to learn new methods for different tasks. Healthcare providers do not spend much time in learning software applications so this concept should efficiently be adopted in HIT applications.

Minimizing Cognitive Load: This principle is indeed essential for healthcare information systems that are usually based on extensive information management and presentation. Healthcare providers work under time constraints and generally use these systems while communicating with patients, so it is certainly required that cognitive load on such users should be minimum. It is appropriate to display every essential information to the user to achieve this principle but this should be done efficiently so that the principle of 'Simplicity' is not affected.

Efficient Interactions: There are multiple approaches and patterns that support efficient interactions however examples of some techniques that directly promote this idea in HIT applications have been discussed. The steps

to complete a task should be minimum or we can say that no additional step should be there to complete a task. Moreover, shortcuts and auto complete options also help in achieving this principle.

Forgiveness and Feedback: Forgiveness can be best achieved when users can explore an application without fear of problematic results and calamity. Moreover, user's mistakes should be efficiently handled by the application and it should not lead to appalling consequences. The functionality to undo some task greatly helps with this principle. The concept of forgiveness should be supported by proper feedback that helps in avoiding a number of problems. Feedback also helps in increasing user's trust by telling that a task has been completed successfully. Although the principle of forgiveness and feedback is required for all applications, but its importance is really high in HIT applications where user errors can have dire consequences.

Effective Use of Language: Using appropriate language and terminologies is essential for every application, but its significance increases with a notable magnitude when it comes to healthcare systems. Terminologies used in such systems must comply with the context of work and should be helpful and meaningful. System related terms should not be displayed on the interface. This concept has been discussed in different set of principles that focus on improving usability.

Effective Information Presentation: This principle covers a number of aspects starting from proper selection of colors to density of information on screens. In HIT applications these factors should meet the requirements of consultant aesthetic concepts as well as screen dimensions. Certain colors and symbols have specific meanings, so they should be used accordingly. Moreover, the information presented on the interface should be readable enough.

Preservation of Context: This principle implies that the interface should remain consistent during the process of a task completion. Interruptions in visual focus of the user should be limited as it is difficult to refocus on the actual task. It has been submitted that dialog box is the most critical component that hinders preservation of context and shifts the focus of the user. Efficient ways of interacting with the user should be introduced in the applications to achieve this principle.

Seffah and Metzker [24], in their research discuss the gap between usability and software engineering. They make the point that in recent past, many techniques for user-centered design (UCD) have been developed by Human Computer Interaction (HCI) community. However, these techniques are merely used by software development organizations as they are difficult to understand and use. An important point here is that the methods developed to enhance usability are themselves underused. A strong reason behind it is that these techniques are developed independently by HCI community and they do not connect well with software engineering life cycle concepts. The research gives stress on the development of additional tools that assist software developers in making UCD fit into their developed design and project context. Interface designing skills are neglected by managers and software developers, making it a basic reason for less use of UCD methods. Usability techniques and software engineering disciplines should learn from each other and interact to facilitate their integration. Learning from these concepts, it can be focused in earlier stage of development that usability architecture is in connection with software design.

Gandhi et al. [25] reported in their research that electronic prescription systems definitely reduce the number of medication errors because of their proper checks and data availability. But it is not the case that they also assist the provider to cope with adverse reactions problems and others. The reason behind is that providers are not using allergies and adverse reaction modules properly. eRx still needs many developments, and once those are in place, providers as well as patients can benefit more through these systems. Although, this research was carried out with diverse data taken at different times and by different physicians, yet it cannot act as a concrete research as physician's behavior is definitely linked to the demographics. Prescribers at different demographic locations have different tendencies of prescribing with error and not using a system properly. Physicians of the areas where prescribing structure is fully electronic and well developed, properly know the process and cause fewer prescribing errors. Thus, a research carried out in a particular country cannot be equally useful for all. However, a specific point can be picked from this study, that while designing all connected modules of a prescribing system should be given due importance. Moreover, prescriber's training and variations in different prescribing applications should also be considered.

Sarnikar and Murphy [26] proposed a usability analysis framework for the domain of Healthcare Information Technology (HIT). The research focused on the fact that organizations are investing in HIT to improve quality of

care and to reduce healthcare costs. So it is required that analysis should be done to evaluate the investments and study what impacts they are having. A framework is proposed to identify, classify and prioritize usability errors. The framework helps in dealing with technology centric issues, and finding out their causes and effects. Moreover, it also specifies some basic requirements such framework should satisfy. Firstly, it should be reliable across different systems, platforms and coders. It should be flexible enough to operate in different testing situations. The framework should be able to prioritize errors on the basis of their possible outcomes and consequences. In addition, it should be customized to the HIT context and should specifically highlight healthcare and clinical tasks. It should focus all usability problems and observed errors in order to deal with usability issues in a better way. Though, the framework has been designed while considering many angles of HIT and software usability, but it has some extent of idealism also. It is very difficult to address each and every usability problems in clinical context. Emerging technologies tend to introduce new kinds of problems also, thus such a framework must be highly flexible to adopt changes.

Chapter 3

Proposed Process

The discussed studies and research papers focus on different aspects of software usability and provide a help for a basic learning of usability aspects in healthcare applications. A very important lesson is that software design should be well connected with usability concepts. In the coming chapters, we will focus on proposing our process which combines software engineering activities with usability tasks and evaluation techniques. Moreover, we will also present an account of adoption of this process in practical projects and will present gathered results from the projects.

In this chapter, we propose a new model as shown in Figure 3.1, that focuses on usability activities and evaluation methods. We present a comprehensive development process which can be adopted practically to develop software applications. It is developed on the basic structure of UCD, which is adopted by UEL and UDP as well. Here we discuss our process on the basis of its salient characteristics and features that include integration of usability engineering (UE) and software engineering (SE) activities with constant usability evaluation throughout the process. This reduces the chances of usability issues in later stages as well as after different levels of the process. Moreover, it exploits the benefits of user-focused evaluation techniques, which confirms constant user involvement in the development process. Usability activities and evaluation techniques are split over phases of Requirements Analysis, Design, Development, Validation and Deployment.

We have separated the figures of phases involved in the process and have presented independently. However, it must be noted here that these phases are combined in the process and have strong association between them. Here we discuss these phases and the involved tasks, artifacts and steps. A graphical key for notation used in the process is given in Figure 3.2.

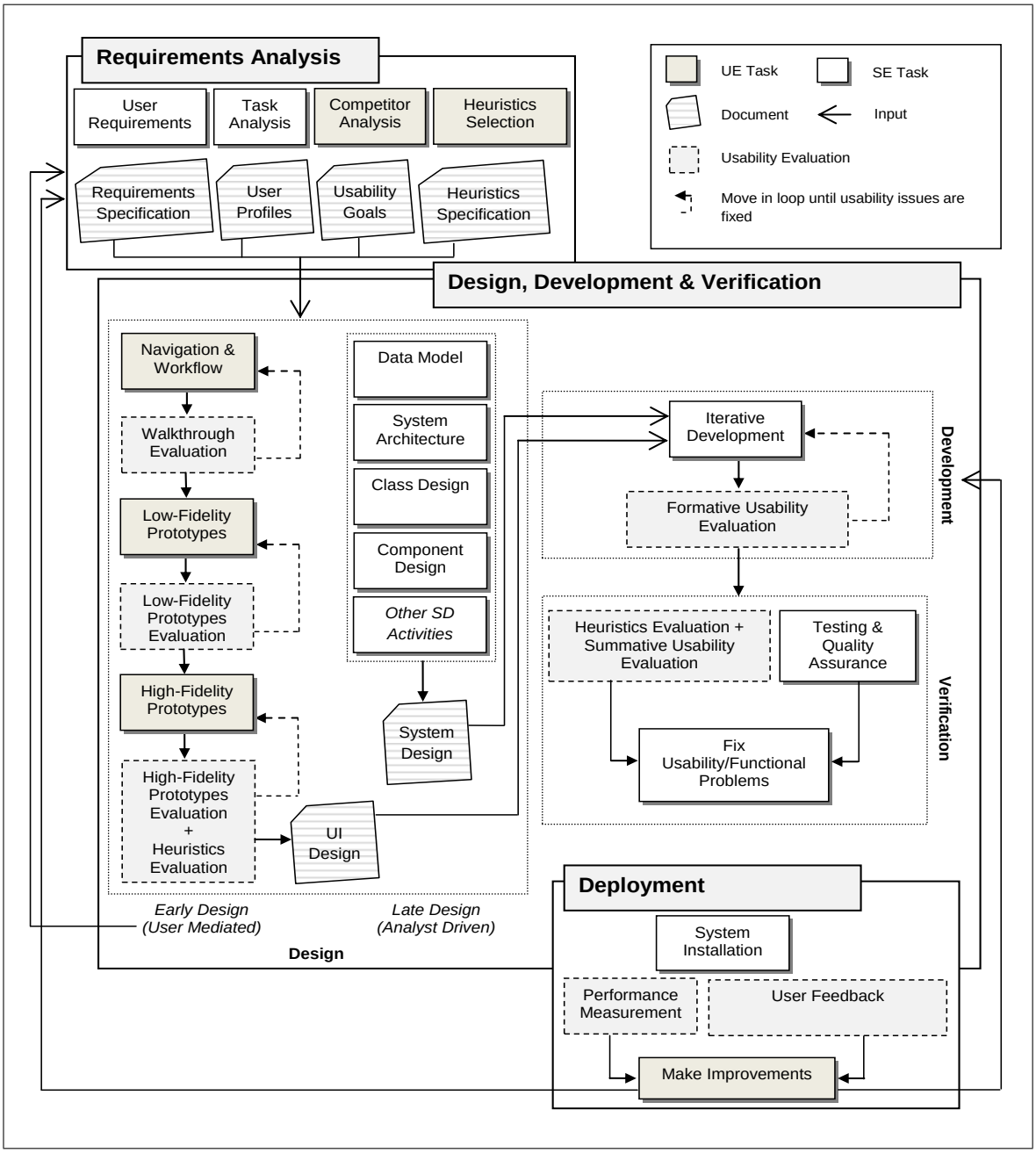


Figure 3.1: Proposed Process

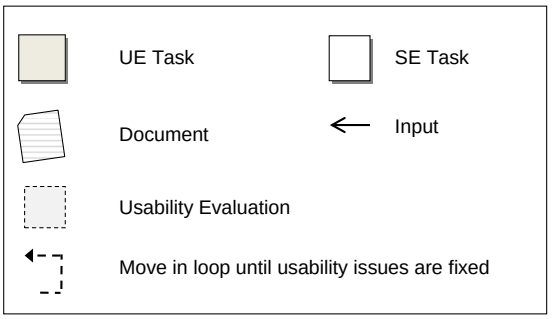


Figure 3.2: Notation used in the proposed process

3.1 Requirements Analysis

The first phase (Figure 3.3) deals with user requirements, their tasks and goals on the basis of which user profiles and usability goals are determined. These activities are very much related to requirements analysis phase of some usability processes, however, we strongly recommend doing a competitive analysis to gain knowledge about applications of similar domain and genre. This helps in setting high standards of usability as well as getting guidelines for a better design. Moreover, we also advocate doing proactive field study in order to gain better knowledge of user tasks, work environment and user capabilities. We support specifying the requirements through use cases as they are from user's viewpoint and are better understood by software engineers as well. The documented artifacts of this phase include user profiles, usability goals, heuristics specification and requirements specification documents.

In addition to other activities, we suggest selecting set of heuristics according to the domain of software being developed. Although, the set of heuristics developed by Jakob Nielsen [16] are considered as most famous and widely used, however, it has been proved through different studies [27] that these heuristics do not always help efficiently in every domain. So, it is required that heuristics according to specific domain should be selected and analyzed accordingly in evaluation process. A heuristics specification document should be created that defines heuristics and also specifies which heuristics should be evaluated in design stage and which one after development. As certain principles may only require prototypes for evaluation, while

others may require a finished application. A number of new set of usability heuristics have been presented for specific domains like healthcare information technology [23], video games [28], augmented reality applications [29] etc. Moreover, a methodology to create heuristics for specific applications (as discussed earlier) has also been presented [17] which presents defined stages for establishing usability heuristics.

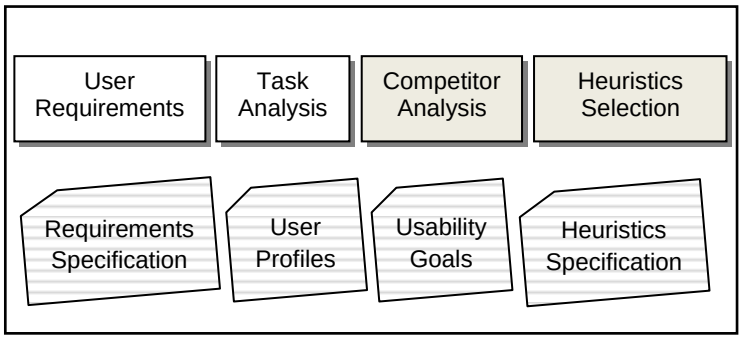


Figure 3.3: Requirements Analysis Phase

3.2 Design Phase

The design phase (Figure 3.4) is divided into two sub phases which are UI design and system design, the former of these is user focused while the later is analyst/software engineer driven. We consider UI design phase as most critical, and this is where most of the usability issues are uncovered and fixed at a very early stage. It must be noted here that this phase helps in dealing with changing requirements in an efficient way. Requirements phase can be revisited with the changes in requirements during this phase. The involvement of user in usability evaluation methods of this phase confirms that quite a few new or changed requirements are exposed at an early stage which reduces the risk of changed requirements later in the process.

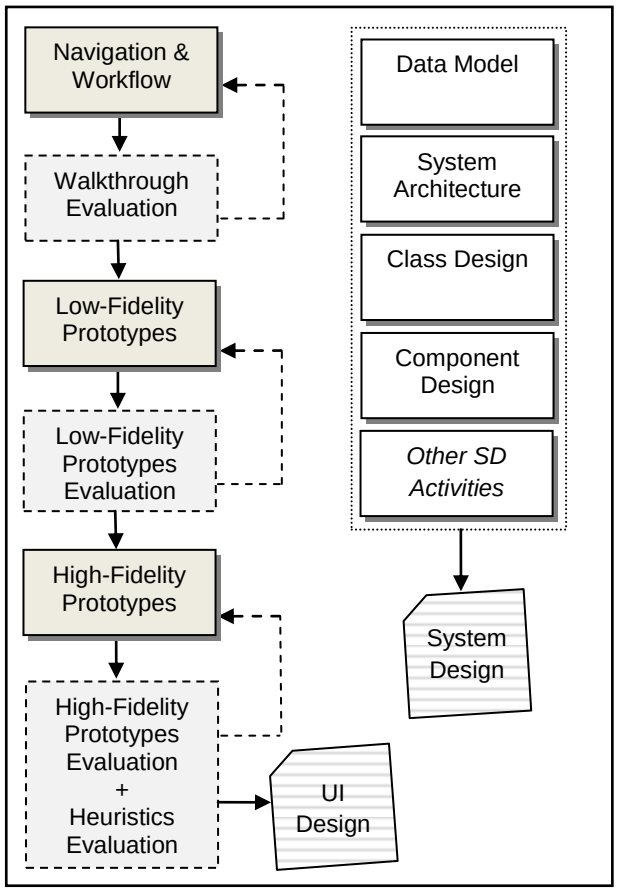


Figure 3.4: Design Phase

3.2.1 UI Design

Initially, a workflow of the system is designed which represents an overall concept of navigation. It is important to specify a workflow at this stage as users have predetermined ways of performing certain tasks. A workflow can be created on paper which is than evaluated by users through walkthrough evaluation. Users go through the paper based navigational flow to identify any problems in it, and designers fix them accordingly.

The next step is to design low fidelity prototypes on the basis of user requirements, usability goals and defined navigation. We recommend using paper mockups for this as they take less time and are easy to modify in eval-

uation sessions with the user. For usability evaluation of these prototypes, we recommend a tailored Coaching Method where users attempt to perform tasks by using the mockups and ask questions from UI designers or usability coaches. Usability experts or UI designers analyze the users and determine the areas which are problematic. This step is very much helpful in uncovering new and changed requirements. We discourage creating high fidelity prototypes without performing this step, as modifying them again and again increases design cost.

Designing high fidelity prototypes focuses on detailed aspects of user interface where working mockups are constructed that represent the actual software application to be developed. Representative data should also be used to get actual flavor of usage. For evaluating high fidelity prototypes a usability evaluation should be carried out along with which evaluators should perform heuristics evaluation according to heuristics specification document created in requirements analysis phase. Users participate in usability evaluation of this step as well, which ensures removal of a number of usability issues.

As mentioned earlier, the UI design phase and continuous usability evaluation involved in it, ensures that major usability issues are screened out at a very early stage. User-focused evaluation techniques confirm constant user involvement. This also helps in creating real expectations from the software application being developed and makes learning process easy once the software is deployed. Moreover, low and high fidelity prototypes are commonly used by software engineers in requirements and design activities, so using them instead of less familiar techniques helps in acceptance of our method.

A user interface specification document is created at the end of this phase which contains workflows and high fidelity prototypes, to be used in development stage. It is necessary to document usability issues as well along with adopted solutions.

3.2.2 System Design

The proposed process considers that many of the usability achieving approaches mention tasks that just focus on improving usability. They do not provide any steps for general software engineering concepts. We have tried to cover this lacking and included software engineering tasks and activities as well. System design phase is also amongst other software engineering activities and steps involved in our process.

In this phase, analysts and software engineers design the system attributes like detailed data models, class designs, component designs and others. This phase can be modified according to the project being developed. A detailed system design document is obtained after this phase which along with the user interface specification acts as input for iterative development phase.

3.3 Development Phase

The software is developed in iterations, after each of which formative usability evaluation is carried out (Figure 3.5). Usability experts figure out any usability problems during user-oriented evaluation, which are then fixed accordingly.

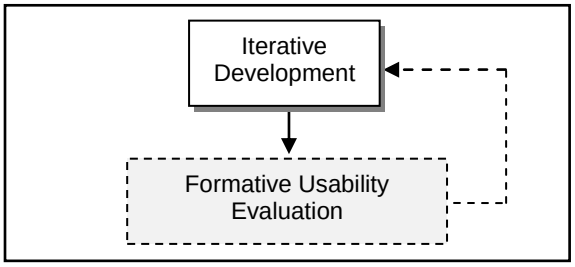


Figure 3.5: Development Phase

3.4 Verification Phase

Throughout the process usability is evaluated by a number of techniques, however, it is essential to verify other aspects of the application as well. Usability approaches studied in literature do not provide a method to include quality assurance techniques in the process. They present a general idea of evaluation; either it is related to usability or other software engineering verification techniques. However, we include software quality assurance techniques in this phase (Figure 3.6) to assure that the software application fulfills all user requirements apart from usability aspects as well. Along with general techniques, usability is evaluated at this stage too in a detailed manner. A summative usability evaluation is carried out which evaluates usability of the application in a detailed manner and specifies how much the software is

usable. Moreover, according to heuristics specification document (created in requirements analysis phase), usability features that are to be evaluated in the developed application should be analyzed at this stage. Issues related to usability and others are fixed and the software is deployed after verification.

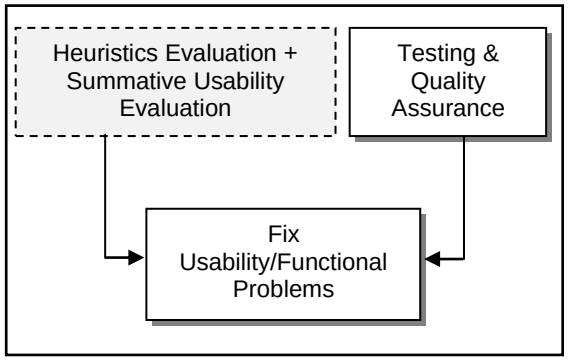


Figure 3.6: Verification Phase

3.5 Deployment Phase

Deployment can be done in incremental way or otherwise, depending upon the nature of the product. Performance measurement evaluations should be carried out as they give quantitative data according to usability goals. Results from performance measurement and user feedback etc should be used to improve usability of the software and if required, development or requirement phases could be revisited (Figure 3.7). In case of users who have been directly involved in UI design and usability evaluation, it gets easy to use the software in better way. Moreover, already developed realistic expectations result in higher levels of satisfaction.

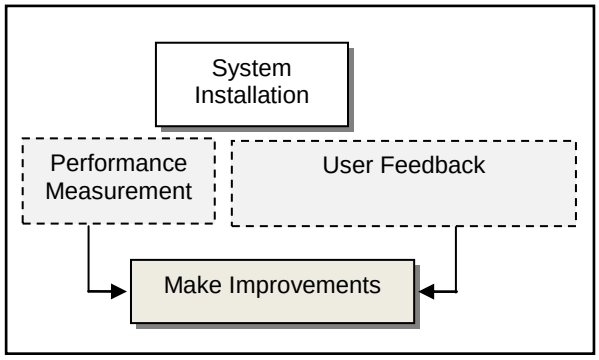


Figure 3.7: Deployment Phase

Chapter 4

Evaluation

In this chapter we present the scenarios in which our proposed process was implemented. Multiple real-life software development projects have been taken as scenarios, in order to present reliable and concrete results. To prove the proposed hypothesis, different perspectives of the process were tested across these real-life projects. As mentioned earlier in Chapter 1, the hypothesis for this work is as follows:

- By adopting the proposed process, we can achieve higher levels of usability.
- The proposed process identifies changing requirements, and thus helps to deal with them.
- Usability issues are uncovered at very early stages of development, by applying the proposed process.
- Time required to perform additional usability activities is negligible.
- It is easy to adopt the proposed usability oriented process.

So, the null hypothesis for this work is as follows:

- By adopting the proposed process, we cannot achieve higher levels of usability.
- The proposed process does not identify any changing requirements.
- Usability issues are not uncovered, or uncovered at late stages of development.
- Time required to perform additional usability activities is significant.
- It is not easy to adopt the proposed usability oriented process.

4.1 Methodology

To evaluate the hypothesis, different software applications were developed by using the proposed process. Following are the software development projects that implemented the process:

Personal Health Record Insurance Module (P1): An application for insurance plan providers to manage offered insurance services. General users can select an insurance plan according to their specific requirements.

Home Health Agency (P2): An application for nurse practitioners that provide health services at patient's home.

Electronic Prescribing Application (P3): An application for doctors to create and manage prescriptions.

Restaurant Booking Application (P4): A smart phone application that allows its users to book tables in restaurants and place their order.

The software applications were of varied scope, complexity and teams that developed these projects were also of different size. To prove all points of hypothesis, particular projects were developed using full process (FP), partial process (PP) and null process (NP). This diverse pool of projects helped in getting reliable data regarding the proposed process. An overview of these applications is shown in Table 4.1.

Table 4.1: Overview of software projects

Project	Process	Team Members	Platform
PHR Insurance Module (A)	NP	2	Windows
PHR Insurance Module (B)	FP	2	Windows
Home Health Agency	FP	4	iOS
Electronic Prescribing Application	PP	2	Android
Restaurant Booking Application (A)	NP	1	Android
Restaurant Booking Application (B)	PP	1	Android

Certain projects were developed without using the proposed process (NP), while the same projects (with similar requirements) were developed using the process partially (PP) (i.e. up to the stage of user focused design), or completely (FP), in order to present a better comparison.

The mentioned software applications were developed by individuals and teams having varied capabilities and skills. A summary of subjects involved in development with respect to mentioned projects is given in Table 4.2.

Table 4.2: Characteristics of subject individuals

Sr.	Individual	Gender	Project	Work Experience
1.	I1	Female	P1	2 year
2.	I2	Female	P1	less than 1 year
3.	I3	Male	P1	2 years
4.	I4	Female	P1	1 year
5.	I5	Male	P2	3 years
6.	I6	Male	P2	2 years
7.	I7	Male	P2	2 years
8.	I8	Male	P2	2 years
9.	I9	Male	P3	2 years
10.	I10	Male	P3	2 years
11.	I11	Female	P4	1 years
12.	I12	Male	P4	2 years

4.2 Variables

This section presents the variables regarding which data was captured during the development of mentioned software projects. Different aspects and characteristics were recorded to cover all aspects of the hypothesis. Following are the variables that were observed and measured in the subject projects.

4.2.1 Usability Problems

Usability problems indicate the quality of a software application. The finished product should contain only a few or ideally no usability problems. This can be attained if usability problems are uncovered and eliminated earlier in development process. The records of usability problems, their description, number of occurrences and stages at which they are found will help us in determining that either or not the proposed process helps in uncovering usability process in earlier stages of project development.

4.2.2 Changes in Requirements

Stable requirements are considered ideal in software development; however, changes in requirements are always there in almost every software project. These changes have a direct effect on cost, quality and schedule. As the hypothesis proposes that the process will help in identifying changes in requirements, so recording the changes with respect to the stages at which they are identified will help in analyzing the hypothesis.

4.2.3 Development Time

The proposed process focuses on spending time in usability related activities throughout the process. Participants were requested to carefully record the time they spent on different activities of the process. Recording the time spent in usability activities as well as in the overall process will help in investigating the effect of usability activities on overall development time. It will help on determining that either or not an early usability focus helps in reducing overall development time.

4.2.4 Process and task complexity

For measuring the complexities of particular tasks involved in the process, and the whole process itself, we asked the participants who used the proposed process in their projects. Participants had to rate the complexity of the process and its particular tasks on a scale of 1-5.

4.2.5 Usability Level

To evaluate usability of the projects included in the evaluation of this thesis, we used Single Ease Question (SEQ) for task level satisfaction, and System Usability Scale (SUS).

4.2.5.1 Single Ease Question (SEQ)

Single Ease Question helps in determining how easy or difficult it is for users to perform certain tasks while using a system. It is a scale that focuses on different levels of difficulties associated with tasks. Although, SEQ is a simple method for measuring usability of a system, yet it is found to be equally well or better [30] [31] than other standard questionnaires for task difficulty.

4.2.5.2 System Usability Scale (SUS)

System Usability Scale was originally developed by John Brooke [6], which is a ten points questionnaire that evaluates effectiveness, efficiency and learning aspects of a system. This scale can be used on small sample sizes and is considered as reliable and valid [32]. Comparisons can be made on the basis of SUS scale, so it is definitely helpful in comparing usability levels of subject projects. The standard SUS questionnaire is given in Appendix A.

4.3 Data

This section presents the data captured during the development of the subject projects. The data was captured according to the variables discussed in the previous section. In order to evaluate usability in different phases of the proposed process, at least five (5) users participated in the process activities, as these much users are sufficient according to Jakob Nielsens mathematical model of usability problems [33].

For usability problems, changes in requirements and development time, comparable results were required, so this section shows the data related to these variables in general software development phases i.e. Analysis, Design, Development, Testing and Deployment. However, it must be noted that usability problems and changes in requirements were uncovered during specific evaluation activities of the proposed process, details of which can be found in Appendices.

4.3.1 Usability Problems

Tables 4.3 and 4.4 respectively indicate the number of problems according to particular applications and with respect to the process used. Usability problems, for the projects which used our proposed process, were uncovered mostly in design phase and a few in later stages. However, usability problems were uncovered at late stages for the projects which did not use the proposed process. Details of problems found in specific applications according to process stage and classified as per Usability Problem Taxonomy [34] can be found in B.

4.3.2 Changes in Requirements

Tables 4.5 and 4.6 respectively indicate the number of problems according to particular applications and with respect to the process used. Like usability

Table 4.3: Number of usability problems

Process Stage	PHR (NP)	PHR (FP)	HHA (FP)	sRx (PP)	Restaurant (NP)	Restaurant (PP)
Analysis	0	0	1	0	0	0
Design	0	7	8	7	2	6
Development	1	2	3	0	5	0
Testing	1	0	1	0	7	0
Deployment	4	0	1	0	4	0

Table 4.4: Number of usability problems

Process Stage	Null Process	Partial Process	Full Process
Analysis	0	0	1
Design	2	13	15
Development	6	0	5
Testing	8	0	1
Deployment	8	0	1

problems, changes in requirements for the projects which used the proposed process were uncovered earlier as compared to the projects which did not use the process.

Table 4.5: Number of changes in requirements according to project

Process Stage	PHR (NP)	PHR (FP)	HHA (FP)	sRx (PP)	Restaurant (NP)	Restaurant (PP)
Analysis	1	1	4	2	4	2
Design	0	8	4	2	1	6
Development	1	1	3	0	6	0
Testing	4	1	4	0	9	0
Deployment	3	0	1	0	3	0

4.3.3 Development Time

Table 4.7 shows the expected and actual time spent in different case study projects. Although, additional time was spent in design phase for the projects which used the process, yet the projects completed in expected time. Significant time was saved in development as major problems and changes in requirements were already dealt with. However, the projects which did not

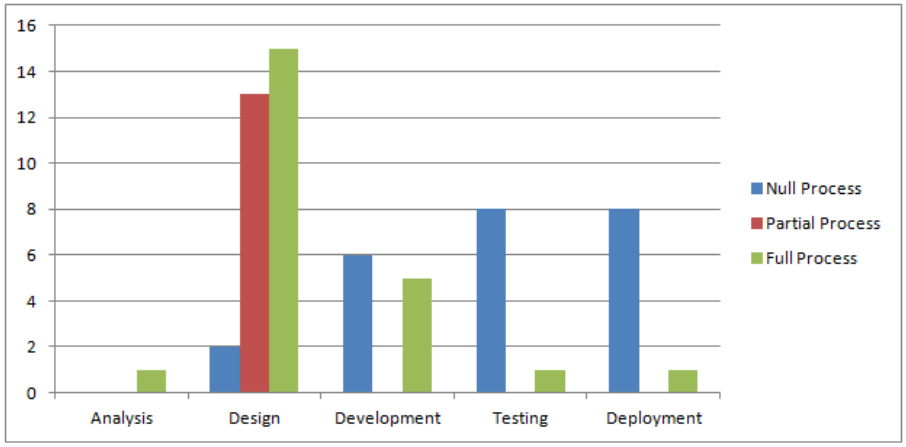


Figure 4.1: Usability Problems

Table 4.6: Number of changes in requirements according to process phases

Process Stage	Null Process	Partial Process	Full Process
Analysis	5	4	5
Design	1	8	12
Development	7	0	4
Testing	13	0	5
Deployment	6	0	1

use the proposed process faced difficulties in dealing with changes in requirements and other problems at later stages of the process. It has also been observed from the case studies that no surprising feedback was received from the users after development iterations, which helped in following the expected schedule of development. Details of development time according to specific phases of the process can be found in Appendix E.

Table 4.7: Development time of different projects in Man Hours

Project	Expected Time	Actual Time
PHR (NP)	64	62
PHR(FP)	64	58
Home Health Agency (FP)	233	217
Restaurant Booking (NP)	368	534

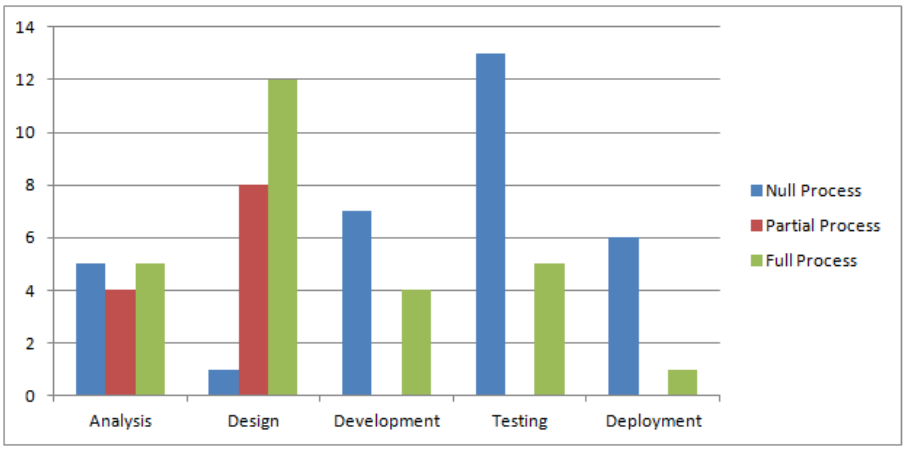


Figure 4.2: Changes in Requirements

4.3.4 Process and task complexity

10 users rated the complexity of particular tasks and of the whole process on a scale of 5. Table 4.8 shows the average complexity of tasks and the process.

4.3.5 Usability Level

To evaluate usability of the projects included in the evaluation of this thesis, we used Single Ease Question (SEQ) for task level satisfaction, and System Usability Scale (SUS).

4.3.5.1 Single Ease Question (SEQ)

Table 4.9 shows the average task level satisfaction calculated from SEQ for different projects. This data has also been represented in Figure 4.3. Average score of SEQ is around 4.8 to 5.1 [35].

4.3.5.2 System Usability Scale (SUS)

Table 4.10 shows the average System Usability Scale obtained from user feedback of different projects. SUS scores of different case study projects have also been represented in Figure A.1. Average score of SUS is 68 [32].

Table 4.8: Task Complexity (1 - Very Difficult, 5 - Very Easy)

Task	Complexity
Proactive Field Study	3.10
Task Analysis	3.00
Competitor Analysis	3.10
Navigation and Workflow	4.40
Usability Evaluation of Navigation and Workflow	4.10
Low-Fidelity Prototypes	4.40
Usability Evaluation of Low-Fidelity Prototypes	3.70
High-Fidelity Prototypes	4.10
Usability Evaluation of High-Fidelity Prototypes	3.10
Iterative Development according to High-Fidelity Prototypes	3.70
Heuristics Evaluation	3.60
User Feedback	3.40
Task Average	**3.64**
Whole Process (from participant's feedback)	**3.70**

Table 4.9: Average SEQ score of case study projects

Project	SEQ
PHR (NP)	4.55
PHR (FP)	5.67
Home Health Agency (FP)	5.90
Restaurant Booking (NP)	4.10

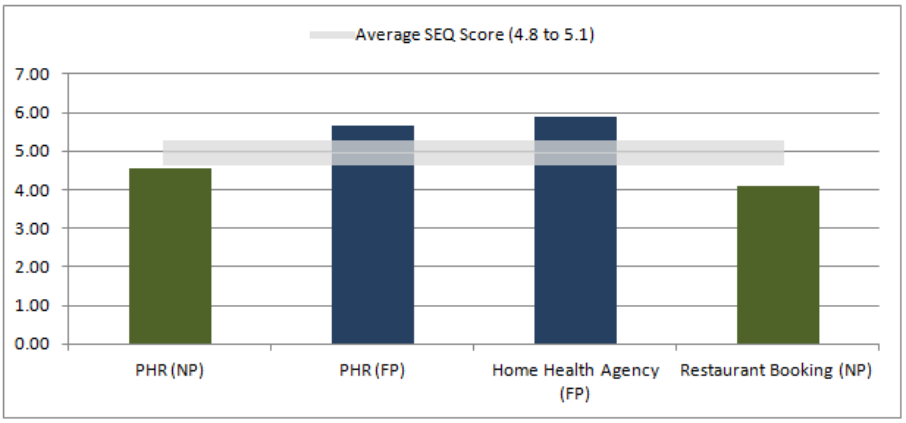


Figure 4.3: Average SEQ score of case study projects

Table 4.10: Average SUS score of case study projects

Project	SUS
PHR (NP)	51.25
PHR (FP)	69.25
Home Health Agency (FP)	81.00
Restaurant Booking (NP)	59.50

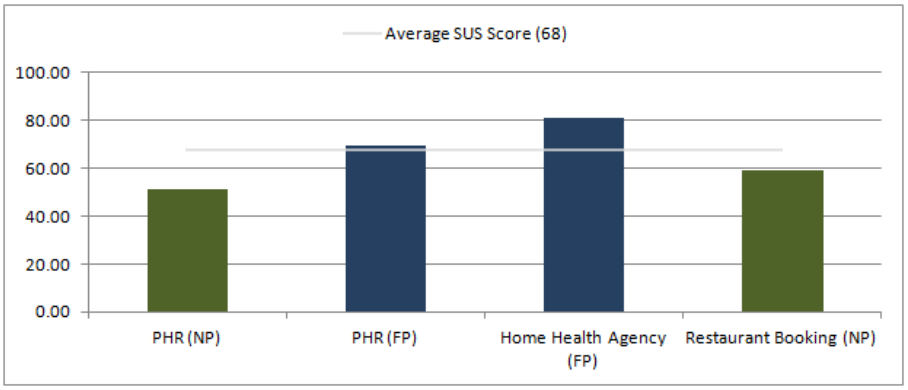


Figure 4.4: Average SUS score of case study projects

Chapter 5

Results and Findings

In the previous chapter, data captured from different case study projects has been presented. The data helped in analyzing the proposed hypothesis of this work. The results that can be drawn from the presented data are presented in this chapter.

The analysis of usability problems shows that the adoption of the proposed process helped in fixing usability problems in early user focused design. For these projects, most of the usability problems were uncovered and fixed in design phase and a very few usability problems were left in later stages. However, usability problems were addressed very late in the projects which did not apply the proposed process. Resultant software applications were also left with usability problems in cases where the process was not used.

Subject projects which followed the proposed process were not troubled by changes in requirements at later stages of development. Most of the changes in requirements were uncovered in the design phase for these projects. However, the projects which did not follow the proposed process had to deal with changes in requirements in later stages of software development as well as after deployment.

The participants of different case studies who used the proposed process have rated the complexity of the process and its tasks. If we analyze the feedback of the participants it can be concluded that in general the participants did not find the process much difficult as the average rating is 3.70 (1 - Very Difficult and 5 - Very Easy), which is significantly towards the easy side. Although, the participants did not faced much difficulties in performing specific tasks of the process, however, it can be analyzed from the data that performing proactive field studies, analyzing user tasks, doing a competitor

analysis and getting feedback from the users are among the tasks that were rated a bit low as compared to other tasks. It can be realized from this that certain professionals might be required for performing these tasks.

The relation of adoption of the proposed process with development time can also be analyzed from the presented data. Although, it seems that additional usability tasks involved in the process may impact the overall development time. However, results from case studies show that even if the participants spent additional time in performing usability activities earlier in the process, yet the overall development time was not affected. The major reason behind this is the time saved in development phase. The subject projects which did not apply the proposed process had to suffer from requirement changes and other issues in later stages of project development, which led to additional time spent.

If we analyze the results obtained from standard usability questionnaires, we can conclude that the case studies for which the proposed process was used, got higher levels of usability and of user satisfaction at task level. The projects that did not apply the proposed process got significantly low levels for these metrics.

Chapter 6

Conclusions and Future Work

This chapter presents a conclusion to our work giving a summary of the proposed method, its evaluation and the results. It also presents an account of research areas and lines that this work leaves open for future studies.

6.1 Conclusions

In this work we presented a complete software development process that focuses on merging usability activities and evaluation methods in a general arrangement of software development activities. The proposed process aims to bridge the gap between the disciplines of software engineering and usability engineering.

Through software development case studies, we have proved that the presented process helps in handling usability issues and changes in requirements in early stages of development Moreover, it can be adopted to develop software applications with better levels of usability. Individuals, who used the proposed process in their software projects, have rated it as quite easy and have not overall spent additional time for completing their projects.

6.2 Future Work

This work generates a number of research lines for future work. In this section we present an account of the areas on which we intend to work on future and those that are left open for further research.

For the evaluation of this work we selected a few variables for which data was gathered from different software development case studies. In future, we

intend to analyze relation of the proposed process with more variables related to software development projects. Possible examples of such variables are size of teams, domain of projects, application platform and some others.

From the obtained results an indication has been received that certain tasks in the process might require specific professionals in order to perform them in a better way. A study on structure of teams required for this process may be of great assistance in practical implementation of the proposed process in software development industry.

Different usability evaluation methods were discussed in this thesis, however, it was not specified that which method is most suitable for specific tasks in the process. Effect of different usability evaluation methods with respect to process tasks should be studied and it should be advised that which usability evaluation technique is most suitable for particular tasks.

Moreover, presenting ways to adopt or modify the proposed process for software applications that cannot be anticipated in early design process, is another future research line. One of the examples of such applications is the video games for which interactive user interface cannot be actually experienced through prototyping. Such an approach has been presented in [36] to ensure usability, playability and effectiveness of video games. Our proposed process can also be adopted as a baseline to present video games specific usability oriented development process.

References

[1] Deborah J Mayhew. The usability engineering lifecycle: A practitioners guide to user interface design, 1999.

[2] Bengt Göransson, Jan Gulliksen, and Inger Boivie. The usability design process–integrating user-centered systems design in the software development process. *Software Process: Improvement and Practice*, 8(2):111–131, 2003.

[3] T Jokela. The kessu usability design process model. *University of Oulu,[online], Available ww. tol. oulu. fi/˜ tjokela/KESSUUDProcessModel2*, 1, 2004.

[4] Andy Whitefield, Frank Wilson, and John Dowell. A framework for human factors evaluation. *Behaviour & Information Technology*, 10(1):65–79, 1991.

[5] Adrian Fernandez, Silvia Abrahão, and Emilio Insfran. A web usability evaluation process for model-driven web development. In *Advanced Information Systems Engineering*, pages 108–122. Springer, 2011.

[6] BrookeJohn. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189:194, 1996.

[7] Alain Abran, Adel Khelifi, Witold Suryn, and Ahmed Seffah. Usability meanings and interpretations in iso standards. *Software Quality Journal*, 11(4):325–338, 2003.

[8] Whitney Quesenbery. What does usability mean: Looking beyondease of use'. In *Annual Conference-Society for Technical Communication*, volume 48, pages 432–436. UNKNOWN, 2001.

[9] *ISO. 9241. Ergonomic requirements for office work with visual display terminals: Part 11.* 1998.

[10] Jakob Nielsen. Usability 101: Introduction to usability. *Jakob Nielsen's Alertbox, August*, 25, 2003.

[11] Institute of Electrical and Electronics Engineers. *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries*. 1990.

[12] Bengt Göransson, Jan Gulliksen, and Inger Boivie. The usability design process–integrating user-centered systems design in the software development process. *Software Process: Improvement and Practice*, 8(2):111–131, 2003.

[13] ISO13407 ISO. 13407: Human-centred design processes for interactive systems. *Geneva: ISO*, 1999.

[14] Jakob Nielsen. *Usability engineering*. Access Online via Elsevier, 1994.

[15] Jakob Nielsen. Usability inspection methods. In *Conference companion on Human factors in computing systems*, pages 413–414. ACM, 1994.

[16] Jakob Nielsen. 10 usability heuristics for user interface design, 2013.

[17] Cristian Rusu, Silvana Roncagliolo, Virginica Rusu, and Cesar Collazos. A methodology to establish usability heuristics. In *ACHI 2011, The Fourth International Conference on Advances in Computer-Human Interactions*, pages 59–62, 2011.

[18] R Ramli and A Jaafar. e-rue: A cheap possible solution for usability evaluation. In *Information Technology, 2008. ITSim 2008. International Symposium on*, volume 3, pages 1–5. IEEE, 2008.

[19] Regina Bernhaupt, Kristijan Mihalic, and Marianna Obrist. Usability evaluation methods for mobile applications. *Handbook Res User Interface Des Evaluation for Mobile Technology*, 44:745–758, 2008.

[20] Robert A Virzi, James F Sorce, and Leslie Beth Herbert. A comparison of three usability evaluation methods: Heuristic, think-aloud, and performance testing. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 37, pages 309–313. SAGE Publications, 1993.

[21] Ann Doubleday, Michele Ryan, Mark Springett, and Alistair Sutcliffe. A comparison of usability techniques for evaluating design. In *Proceedings of the 2nd conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 101–110. ACM, 1997.

[22] Mohd Naz'ri Mahrin, Paul Strooper, and David Carrington. Selecting usability evaluation methods for software process descriptions. In *Software Engineering Conference, 2009. APSEC'09. Asia-Pacific*, pages 523–529. IEEE, 2009.

[23] Jeffery L Belden, Rebecca Grayson, and Janey Barnes. Defining and testing emr usability: Principles and proposed methods of emr usability evaluation and rating. Technical report, Healthcare Information and Management Systems Society (HIMSS), 2009.

[24] Ahmed Seffah and Eduard Metzker. The obstacles and myths of usability and software engineering. *Communications of the ACM*, 47(12):71–76, 2004.

[25] TK Gandhi, SN Weingart, AC Seger, DL Seger, J Borus, E Burdick, LL Leape, and DW Bates. Impact of basic computerized prescribing on outpatient medication errors and adverse drug events. *Journal of the American Medical Informatics Association*, 9(Suppl 6):S48–S49, 2002.

[26] Surendra Sarnikar and Maureen Murphy. A usability analysis framework for healthcare information technology. 2009.

[27] Llucia Masip, Toni Granollers, and Marta Oliva. A heuristic evaluation experiment to validate the new set of usability heuristics. In *Information Technology: New Generations (ITNG), 2011 Eighth International Conference on*, pages 429–434. IEEE, 2011.

[28] David Pinelle, Nelson Wong, and Tadeusz Stach. Heuristic evaluation for games: usability principles for video game design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1453–1462. ACM, 2008.

[29] Sang Min Ko, Won Suk Chang, and Yong Gu Ji. Usability principles for augmented reality applications in a smartphone environment. *International Journal of Human-Computer Interaction*, 29(8):501–515, 2013.

[30] Jeff Sauro and Joseph S Dumas. Comparison of three one-question, post-task usability questionnaires. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1599–1608. ACM, 2009.

[31] Donna Tedesco and Tom Tullis. A comparison of methods for eliciting post-task subjective ratings in usability testing. *Usability Professionals Association (UPA)*, 2006:1–9, 2006.

[32] Jeff Sauro. Measuring usability with the system usability scale (sus), 2011.

[33] Jakob Nielsen and Thomas K Landauer. A mathematical model of the finding of usability problems. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 206–213. ACM, 1993.

[34] Susan L Keenan, H Rex Hartson, Dennis G Kafura, and Robert S Schulman. The usability problem taxonomy: A framework for classification and analysis. *Empirical Software Engineering*, 4(1):71–104, 1999.

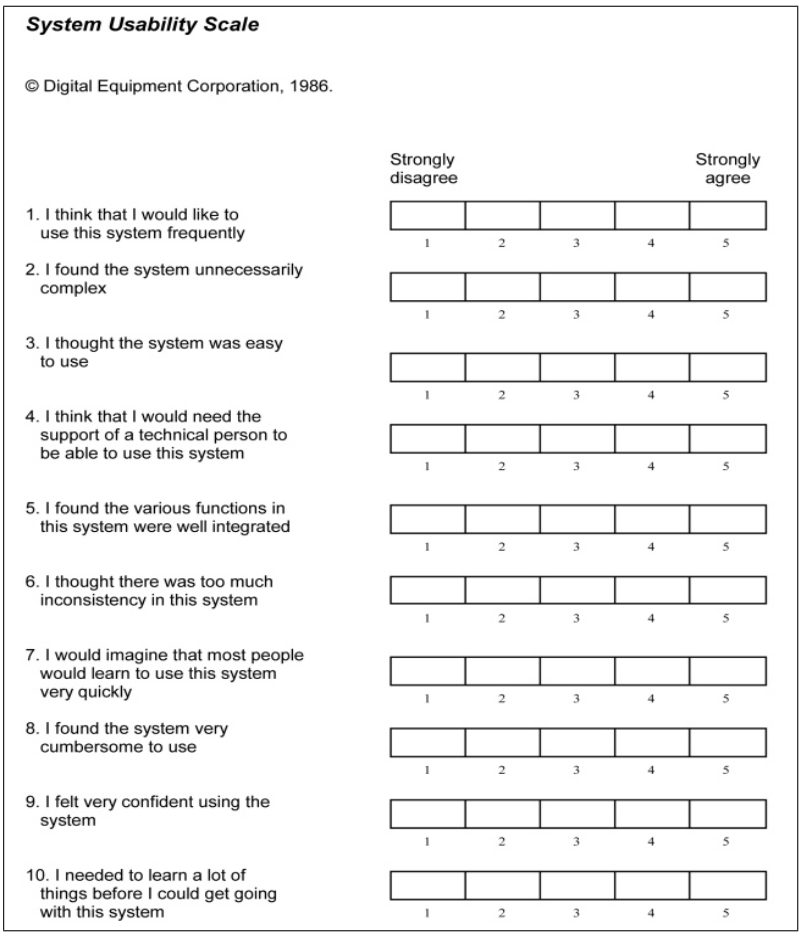
[35] Jeff Sauro. 10 things to know about the single ease question (seq), 2012.

[36] Tanner Olsen, Katelyn Procci, and Clint Bowers. Serious games usability testing: How to ensure proper usability, playability, and effectiveness. In *Design, user experience, and usability. theory, methods, tools and practice*, pages 625–634. Springer, 2011.

Appendices

Appendix A

Standard SUS Questionnaire



System Usability Scale

© Digital Equipment Corporation, 1986.

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently	1	2	3	4	5
2. I found the system unnecessarily complex	1	2	3	4	5
3. I thought the system was easy to use	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5. I found the various functions in this system were well integrated	1	2	3	4	5
6. I thought there was too much inconsistency in this system	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8. I found the system very cumbersome to use	1	2	3	4	5
9. I felt very confident using the system	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Figure A.1: System Usability Scale form (adopted from SUS - A quick and dirty usability scale [6])

Appendix B

Usability Problems

Table B.1: Usability Problems found in Personal Health Record - Insurance Module (Null Process)

Sr.	Problem Description	Artifact Classification	Artifact Outcome	Task Classification	Task Outcome	Stage
1	The size of Edit button is very small.	(Visualness) (Object Appearance)	FC	(Task Mapping) (Interaction)	FC	Development
2	Insurance policy particulars are displayed on a limited space.	(Visualness) (Presentation of Information)	FC	(Task)	NC	Testing
3	The term "Operation" used in insurance policy services is not well known to the users.	(Language) (On-screen Text)	FC	(Task)	NC	Deployment
4	Insurance Policy searching facility is limited.	(Artifact)	NC	(Task Facilitation)	PC	Deployment
5	Additional step for users to start their tasks.	(Artifact)	NC	(Task Mapping)	PC	Deployment
6	There is no reversal option associated with contact modification.	(Artifact)	NC	(Task Facilitation) (User Action Reversal)	FC	Deployment

Table B.2: Usability Problems found in Personal Health Record - Insurance Module (Full Process)

Sr.	Problem Description	Artifact Classification	Artifact Outcome	Task Classification	Task Outcome	Stage
1	Additional step for users to start their tasks.	(Artifact)	NC	(Task Mapping)	PC	Navigation / Workflow
2	No appropriate feedback after saving/ updating records.	(Language) (Other wording) (Feedback Messages)	FC	(Task)	NC	Low-Fidelity Prototypes
3	The terms Operation, Test, Start Date and End Date are not domain specific.	(Language) (On-screen text)	FC	(Task)	NC	Low-Fidelity Prototypes
4	Insurance Policy searching facility is limited.	(Artifact)	NC	(Task Facilitation)	PC	Low-Fidelity Prototypes
5	Inconsistent organization of controls on different screens.	(Visualness) (Object Layout)	FC	(Task Mapping)	PC	High-Fidelity Prototypes
6	Buttons in Account Information section have inconsistent nature.	(Visualness)	PC	(Task Mapping) (Functionality)	FC	High-Fidelity Prototypes
7	The user cannot leave a task once started.	(Artifact)	NC	(Task Facilitation) (User Action Reversal)	FC	High-Fidelity Prototypes
8	Unavailability of searched results filtering.	(Artifact)	NC	(Task Facilitation)	PC	Iterative Development
9	Increase in cognitive load due to limited information display.	(Visualness) (Presentation of Information)	FC	(Task Facilitation)	PC	Iterative Development

Sr.	Problem Description	Artifact Classification	Artifact Outcome	Task Classification	Task Outcome	Stage
1	Unavailability of task list leads to incomplete assistance.	(Artifact)	NC	(Task Facilitation) (Task/Function automation)	FC	Analysis
2	Selecting type of service before selecting a patient causes an unnecessary cycle.	(Artifact)	NC	(Task Mapping) (Navigation)	FC	Navigation / Workflow
3	Selecting task related to patient does not match real life workflow.	(Artifact)	NC	(Task Mapping)	PC	Navigation / Workflow
4	The term 'Came for' is not domain specific.	(Language) (Other wording) (On-screen text)	FC	(Task)	NC	Low-Fidelity Prototypes
5	Patient list status controls are repeated for each patient.	(Visualness) (Object Appearance)	FC	(Task)	NC	Low-Fidelity Prototypes
6	Unnecessary information in patient list.	(Visualness) (Presentation of information)	FC	(Task)	NC	High-Fidelity Prototypes
7	Displaying menu items all the time limits available space.	(Visualness) (Object Layout)	FC	(Task)	NC	High-Fidelity Prototypes
8	Administrator Notes remain empty for most of the cases, and capture unnecessary space.	(Visualness)	PC	(Task)	NC	High-Fidelity Prototypes

9	User requires to type for confirming if the patient is fasting.	(Artifact)	NC	(Task Facilitation)	PC	High-Fidelity Prototypes
10	Directions Panel is not well aligned with route map.	(Visualness) (Object layout)	FC	(Task Facilitation)	PC	Development
11	Using self made UI instead of standard forms creates discomfort for user (e.g. OASIS form).	(Artifact)	NC	(Task Facilitation)	PC	Development
12	Activating directions panel each time, creates an additional step.	(Artifact)	NC	(Task Facilitation) (Task/Function automation)	FC	Development
13	Space for wounds and supplies information remains occupied even if they are empty.	(Visualness)	PC	(Task)	NC	Testing
14	Markers on route map sometimes overload the map.	(Visualness)	PC	(Task Facilitation)	PC	Deployment

Appendix C

System Usability Scale

System Usability Scale gives a single number that represents overall usability of the system. The standard questionnaire contains ten questions that cover different aspects of usability of a system. Users rate the questions on a 5 point scale after which SUS score is calculated. For odd numbered questions the score is scale point minus one, while for even numbered questions the score is 5 minus scale point. The sum of scores is then multiplied by 2.5 to obtain overall SUS score which ranges from 0 to 100. Here, we present details of SUS score for different case study projects included in evaluation of this thesis. (Users 1-10 rated the questions on a scale of 5, 1 - Strongly Disagree, 5 - Strongly Agree). Average score of SUS is 68 [32].

Table C.1: System Usability Scale PHR - Insurance Module (Null Process)

		1	2	3	4	5	6	7	8	9	10
1	I think that I would like to use this system frequently.	3	3	1	1	1	3	3	2	3	3
2	I found the system unnecessarily complex.	3	3	3	2	2	3	3	2	3	3
3	I thought the system was easy to use.	2	2	2	3	3	3	3	4	3	3
4	I think that I would need the support of a technical person to be able to use this system.	3	1	1	2	3	3	2	2	2	2
5	I found the various functions in this system were well integrated.	3	1	2	3	2	3	4	2	3	3
6	I thought there was too much inconsistency in this system.	3	4	2	2	3	3	2	3	2	2
7	I would imagine that most people would learn to use this system very quickly.	1	4	3	3	2	4	3	3	4	2
8	I found the system very cumbersome to use.	2	2	2	3	4	3	3	3	2	2
9	I felt very confident using the system.	1	3	3	2	1	4	3	3	3	3
10	I needed to learn a lot of things before I could get going with this system.	4	2	1	4	4	2	3	1	2	2
	User Score (after applying SUS formula)	32.5	52.5	55.0	47.5	32.5	57.5	57.5	57.5	62.5	57.5
	Total Average	**51.25**									

Table C.2: System Usability Scale PHR - Insurance Module (Full Process)

		1	2	3	4	5	6	7	8	9	10
1	I think that I would like to use this system frequently.	3	2	2	3	4	3	3	3	3	3
2	I found the system unnecessarily complex.	2	2	2	2	3	2	1	4	1	2
3	I thought the system was easy to use.	5	4	5	2	4	4	4	5	4	4
4	I think that I would need the support of a technical person to be able to use this system.	2	2	2	1	4	2	1	3	1	1
5	I found the various functions in this system were well integrated.	5	3	5	4	5	2	3	4	4	3
6	I thought there was too much inconsistency in this system.	1	1	1	2	1	4	2	2	2	2
7	I would imagine that most people would learn to use this system very quickly.	4	4	4	5	3	4	3	3	5	4
8	I found the system very cumbersome to use.	2	2	2	2	3	4	2	3	1	1
9	I felt very confident using the system.	2	4	1	4	5	4	3	3	2	3
10	I needed to learn a lot of things before I could get going with this system.	4	1	3	2	1	2	2	2	3	1
	User Score (after applying SUS formula)	70.0	72.5	67.5	72.5	72.5	57.5	70.0	60.0	75.0	75.0
	Total Average	**69.25**									

Table C.3: System Usability Scale Home Health Agency (Full Process)

		1	2	3	4	5	6	7	8	9	10
1	I think that I would like to use this system frequently.	4	5	5	4	5	4	5	4	5	5
2	I found the system unnecessarily complex.	2	2	2	1	2	2	1	1	1	3
3	I thought the system was easy to use.	4	4	4	4	5	4	4	4	4	4
4	I think that I would need the support of a technical person to be able to use this system.	2	3	1	2	2	3	2	2	2	2
5	I found the various functions in this system were well integrated.	4	4	4	4	4	3	3	4	4	3
6	I thought there was too much inconsistency in this system.	1	1	1	2	2	2	2	1	2	1
7	I would imagine that most people would learn to use this system very quickly.	4	4	4	5	5	4	5	3	5	4
8	I found the system very cumbersome to use.	1	1	1	1	1	2	2	1	1	1
9	I felt very confident using the system.	3	4	5	4	4	4	4	3	3	3
10	I needed to learn a lot of things before I could get going with this system.	2	1	2	2	2	2	1	2	2	1
	User Score (after applying SUS formula)	80.0	82.5	87.5	85.0	85.0	70.0	82.5	77.5	82.5	77.5
	Total Average	**81.00**									

Table C.4: System Usability Scale Restaurant Booking Application (Full Process)

		1	2	3	4	5	6	7	8	9	10
1	I think that I would like to use this system frequently.	2	3	2	1	2	3	1	2	1	3
2	I found the system unnecessarily complex.	3	3	2	2	3	3	3	2	2	3
3	I thought the system was easy to use.	3	4	3	2	3	3	3	2	2	3
4	I think that I would need the support of a technical person to be able to use this system.	1	1	1	2	1	2	2	2	2	1
5	I found the various functions in this system were well integrated.	3	4	2	1	3	3	2	1	3	3
6	I thought there was too much inconsistency in this system.	3	3	2	2	2	3	3	3	2	2
7	I would imagine that most people would learn to use this system very quickly.	4	4	4	2	3	3	4	3	3	3
8	I found the system very cumbersome to use.	2	2	1	2	3	2	3	2	2	2
9	I felt very confident using the system.	4	4	3	2	3	2	3	2	2	3
10	I needed to learn a lot of things before I could get going with this system.	2	2	3	2	2	3	2	1	3	1
	User Score (after applying SUS formula)	62.5	70.0	62.5	45.0	57.5	55.0	50.0	50.0	47.5	65.0
	Total Average	**56.50**									

Appendix D

Single Ease Question

In this chapter, we present the details of task level satisfaction questionnaires as filled by 10 users for each case study project. The responses were against single ease question (i.e. Overall, how difficult or easy was the task to complete?) for specific tasks of software applications. (Users 1-10 rated the tasks on a scale of 7, according to the question, 0 - Failed to perform, 1 - Very Difficult, 7 - Very Easy). Average score of SEQ is around 4.8 to 5.1 [35].

Table D.1: Task Level Satisfaction - Personal Health Record - Insurance Module (Null Process)

		1	2	3	4	5	6	7	8	9	10
1	User login	6	6	7	6	2	6	6	5	5	7
2	View Contact Information (Insurer)	5	6	6	6	5	6	7	7	6	6
3	Modify Contact Information (Insurer)	4	6	5	4	5	5	3	6	4	5
4	View Insurance Policies	5	6	5	6	6	6	3	5	6	6
5	Add Insurance Policy	5	4	1	3	3	2	0	4	0	2
6	Modify Insurance Policy	3	4	4	3	0	3	5	5	0	5
7	View Contact Information (Account Holder)	5	5	5	5	6	7	4	6	6	6
8	Modify Contact Information (Account Holder)	5	3	5	6	6	2	5	6	6	6
9	Search Insurance Policy	2	1	3	3	3	3	2	4	5	1
10	Choose Insurance Policy	1	5	4	5	6	6	5	6	6	6
	User Average	4.10	4.60	4.50	4.70	4.20	4.60	4.00	5.40	4.40	5.00
	Total Average	**4.55**									

Table D.2: Task Level Satisfaction - Personal Health Record - Insurance Module (Full Process)

		1	2	3	4	5	6	7	8	9	10
1	User login	7	7	6	7	6	6	6	6	6	7
2	View Contact Information (Insurer)	6	6	4	5	6	7	7	6	6	6
3	Modify Contact Information (Insurer)	6	6	7	6	7	6	6	6	7	5
4	View Insurance Policies	4	6	5	4	5	5	6	7	6	6
5	Add Insurance Policy	5	5	3	5	7	5	6	6	5	6
6	Modify Insurance Policy	4	4	3	6	6	5	4	6	6	5
7	View Contact Information (Account Holder)	5	6	4	6	5	6	6	7	6	6
8	Modify Contact Information (Account Holder)	6	6	7	6	5	7	6	6	5	7
9	Search Insurance Policy	6	5	7	7	6	4	6	5	6	7
10	Choose Insurance Policy	5	1	1	6	6	7	6	5	5	7
	User Average	5.40	5.20	4.70	5.80	5.90	5.80	5.90	6.00	5.80	6.20
	Total Average	**5.67**									

Table D.3: Task Level Satisfaction - Home Health Agency (Full Process)

		1	2	3	4	5	6	7	8	9	10
1	User login	6	7	7	6	7	6	7	7	7	7
2	View Patient List	7	6	7	7	7	7	7	7	6	7
3	Get Directions	5	6	6	4	7	7	6	6	6	7
4	Get Route Information	6	6	7	6	7	7	5	7	6	7
5	Take Vital Signs	4	5	6	6	6	5	6	6	5	6
6	Fill Visit Documentation	5	5	5	3	3	5	6	5	3	6
7	Enter Wounds Information	6	6	7	6	6	7	6	7	6	6
8	Update Medication Information	6	5	6	6	5	6	7	6	6	4
9	OASIS Form	5	5	6	4	5	5	5	4	5	4
10	Change Patient Visit Status	5	6	6	6	6	6	7	6	7	7
	User Average	5.50	5.70	6.30	5.40	5.90	6.10	6.20	6.10	5.70	6.10
	Total Average	**5.90**									

Table D.4: Task Level Satisfaction - Restaurant Application (Null Process)

		1	2	3	4	5	6	7	8	9	10
1	Task Selection (Welcome Screen)	6	4	6	6	5	6	6	7	7	5
2	Select Restaurant Table	4	4	5	6	6	5	4	7	6	4
3	Select Event Participants	4	5	5	6	4	5	4	6	4	3
4	Provide Event Particulars	2	4	4	5	2	2	2	3	6	3
5	Make Table Reservation	4	4	4	5	3	1	4	3	5	2
6	Check in for Reservation	4	4	5	5	4	3	3	3	1	2
7	Review Restaurant Menu	3	5	5	4	2	2	4	6	4	4
8	Place Order (from menu)	3	5	4	4	4	2	4	6	2	5
9	Bill Payment (multiple options)	3	4	4	4	4	3	4	4	4	5
10	Cancel Table Reservation	4	5	0	5	5	1	5	6	1	4
	User Average	3.70	4.40	4.20	5.00	3.90	3.00	4.00	5.10	4.00	3.70
	Total Average	**4.10**									

Appendix E

Development Time

Table E.1: Development Time w.r.t. process stages in Man Hours

Process Stage	PHR (NP)	PHR (FP)	Home Health Agency (FP)	Restaurant (NP)
Analysis	10	13	48	8
Design	12	19	64	75
Development	33	22	88	400
Testing	6	3	14	48
Deployment	1	1	3	3
Total	**62**	**58**	**217**	**534**