A Usability Maturity Model for Open Source Software

(Thesis format: Monograph)

by

Arif Raza

Graduate Program in Electrical and Computer Engineering

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

THE UNIVERSITY OF WESTERN ONTARIO
School of Graduate and Postdoctoral Studies

CERTIFICATE OF EXAMINATION

Supervisor	Examiners
Dr. Luiz Fernando Capretz	Dr. Ajit Singh
Supervisory Committee	
	Dr. Kamran Sedig
	Dr. Abdelkader Ouda
	Mr. Danny Ho

The thesis by

Arif Raza

Entitled:

A Usability Maturity Model for Open Source Software

is accepted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Date	Chair of the Thesis Examination Board

Abstract

User satisfaction has always been a major factor in the success of software, regardless of whether the software is closed proprietary or open source software (OSS). Although user-centered designs are gaining recognition among the OSS community, many design scenarios still do not incorporate usability as one of their primary goals. Accordingly, many individuals believe that if OSS was more usable, its popularity would increase tremendously. Although there are strong examples of usable open source software, there is still potential to improve the usability of OSS.

The usability assessment of open source projects is an area where relatively little research has been conducted, and, accordingly, the main contribution of this work is a methodology that evaluates the usability maturity of an OSS project. Consequently, we present a usability maturity model that is aimed at usability-related issues for open source projects. The model has been developed in response to a need for measuring the extent to which open source software projects support usability. Specifically, it is intended for assessing and improving the usability aspect in open source software development.

The proposed model examines the relationship between open source projects and their usability aspects. The measuring instrument for the model contains factors that have been selected from four of our empirical studies, which examine the perspectives of OSS users, developers, contributors and the industry. In addition to presenting the usability maturity model, this thesis discusses assessment questionnaires, a rating methodology and two case studies.

Keywords

Open Source Software, Usability, Users, Developers, Contributors, Industry, Maturity Model, Empirical Analysis.

Dedicated To My Parents

They have been the biggest influence in my life.

Acknowledgments

First of all, I am grateful to Allah Almighty for all of His blessings. I am especially thankful for being given the opportunity to carry out my Ph.D. studies. It is my firm belief that this thesis would not have been possible unless I was particularly blessed.

I am extremely thankful for and owe my deepest gratitude to my supervisor, Dr. Luiz Fernando Capretz, whose encouragement, guidance and support throughout all stages of the thesis process enabled me to develop an understanding of the subject. I consider myself fortunate that I had the opportunity to work under his candid supervision.

I owe my sincere gratitude to Dr. Faheem Ahmed of the United Arab Emirates University. He has provided many forms of guidance and tremendous support throughout all the phases of this research work.

Moreover, I would like to show my appreciation to my wife and my children, whose love and reassurance allowed me to return to school and encouraged me through all of the hardships of academia. My sisters and their families are also close to my heart, as I greatly appreciate their moral support and their prayers for my success.

I am particularly thankful to National University of Sciences and Technology (Pakistan), for sponsoring my Ph.D. studies.

I would also like to thank my friends and well-wishers, including Mrs. Linda Munn, Mr. Mark Matsumoto and Mr. Najam Naqvi, who selflessly offered their assistance and supported my endeavor to start a new life in Canada.

Lastly, I offer my regards to everyone who has been a part of this journey and has assisted me in any respect during my endeavors.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations and Symbols

Chapter 1

1 Introduction

The use of free and open source software has increased significantly in recent years, mainly due to the accessibility and availability of the Internet. However, OSS has experienced a number of challenges, including geographically distributed developers, minimal documentation and post release software management. In addition, the experience of end users has become an important issue. With a significant increase in the use of OSS by both individuals and organizations, the level of usability and its related issues must be addressed more thoroughly. However, the usability aspects of open source software cannot be improved unless there are ways to test and measure these aspects. The increasing popularity of open source projects among novice, non-technical users necessitates a usability evaluation methodology. Paulk et al. (1995) maintain that process maturity is the extent to which an organization is able to define, manage, measure, and control a specific process. Accordingly, the main purpose of this research is to study the usability issues of OSS projects from the perspective of different stakeholders. Specifically, the main goals of this research are:

1. To identify the factors that can improve OSS usability.
2. To use the identified key factors as a measuring instrument in presenting the Usability Maturity Model for assessing the usability of OSS projects.
3. To apply the Open Source Usability Maturity Model (OS-UMM) for assessing the current maturity of OSS projects.

Empirical investigation has been conducted to examine the impact of some usability factors on OSS usability. Subsequently, research models have been developed to study and establish the relationship among the usability factors from the perspective of different stakeholders and in terms of OSS usability. Finally, the identified key factors have been used as a measuring instrument to propose the Open Source Usability Maturity Model, OS-UMM. The structural composition of OS-UMM consists of assessment frameworks from four dimensions, which are based on the perspectives of OSS developers, users, contributors, which include architects, designers, developers, testers, and users, as well as the industry. Consequently, the model is used to assess the current maturity of an OSS project by defining an assessment methodology and conducting case studies. Once the assessment results of the individual dimensions are evident, the methodology for evaluating an OSS project's usability maturity profile is utilized as an integral feature of OS-UMM.

Overall, this research work presents a usability maturity model for open source projects. In particular, it provides a methodology for evaluating the current usability maturity of an OSS project. Therefore, this work contributes towards the establishment of a comprehensive and integrated strategy for the usability maturity evaluation of OSS projects. The Usability Maturity Model will assist OSS designers and developers in conducting usability assessments of their projects, thus enhancing their ability to plan improvement strategies.

1.1 Open Source Software and Usability

Open source software refers to software that is equipped with licenses providing current and future users with the right to use, inspect, modify, and distribute modified or unmodified versions of the software to others (Raymond 1999). Raymond maintains that the development culture of OSS, along with the concept of providing free access to the software and its source code, raises the status of OSS to that of a phenomenon.

OSS has influenced almost every dimension of the software development area, thus indicating its significant progress and development. The most successful examples of this influence include the GNU/Linux operating system, the Apache HTTP server, the Mozilla Firefox internet browser, and the MySQL database system. Since OSS systems have neither the physical nor the commercial boundaries of proprietary software, users from all over the world can interact with them. On the one hand, this free access is advantageous, because as increasing numbers of users are able to access OSS, there are more chances of improvement. On the other hand, the aspect and measurement of quality assurance (QA) as well as the post-release management of OSS projects are some of the areas where closed source proprietary software is superior.

The International Organization for Standardization and the International Electrotechnical Commission ISO/IEC 9126-1 (2001) classifies software quality attributes into six categories: functionality, reliability, usability, efficiency, maintainability and portability. Bevan (2009), in referring to the international standards for usability, states that *"these standards provide a great way to integrate usability with quality, but do not help if quality is a low priority in your organization."* Additionally, he identifies four human-centered activities for software design, which include understanding and specifying the context of use, specifying the user requirements, the design solutions and the evaluation.

The users of open source software have free access as well as the ability to modify the source code (Aberdour 2007). Nevertheless, it is likely that usability is the least addressed area in OSS research and development. Traditionally, OSS was designed for technically adept users, thus resulting in a lack of distinction between developers and users. Consequently, usability has never been a top priority for OSS developers. However, OSS is no longer used solely by computer developers; the number of non-technical and novice computer users are growing at a fast pace, highlighting the necessity of understanding and addressing their requirements and expectations (Iivari 2009-a). Accordingly, the usability element of OSS is challenged by the variety of users, who have both technical and non-technical backgrounds as well as cultural diversity, possessing unique needs, expectations and demands. Nichols and Twidale (2005) address another reason as to why usability aspects are neglected in favor of functionality issues; the latter possess more challenges and recognition factors from the community. According to these authors, *"the existence of a problem does not necessarily mean that all OSS interfaces are bad or that OSS is doomed to have hard to use interfaces, just a recognition that the interfaces ought to be and can be made better."*

Iivari et al. (2008) observe that OSS is targeted to the mass population. Underscoring the importance of usability in a company context, they observe that *"typically, the OSS developers do not have knowledge about the non technical users, their tasks and the* *context of use."* Furthermore, they suggest the need for expert opinions as well as realistic user feedback at an earlier OSS design stage. According to Çetin and Göktürk (2008), OSS usability is a multidimensional problem area, particularly due to the fact that it is not a primary goal of OSS projects; OSS developers are not aware of the importance of usability and of users' requirements, and there is a lack of interaction between the developers and the Human Computer Interaction (HCI) Community. Moreover, Hedberg et al. (2007) state that the target users of OSS projects are no longer only co-developers; consequently, OSS systems need to be designed to meet the requirements, expectations and demands of non-technical users.

1.2 Research Questions

The experience of end users has profound impacts on the success of software projects; hence, usability and its related issues has been a key area of research in the open source community. In order to identify precisely where improvement is required, assessment needs to be conducted. OSS is a relatively new software arena, as traditionally, software was produced *"by engineers for engineers"* (Benson et al. 2004). However, with the continual increase in the number of both technical and non-technical OSS users, OSS usability evaluation requires a comprehensive strategy that has not yet been fully explored.

Therefore, the main purpose of this research is to find answers to a series of formulated research questions in order to fill the research gap in the area of open source software usability assessment. The answers to these questions would thus provide a comprehensive methodology for assessing the usability maturity of open source software projects. The following seven research questions compose the theme of this thesis:

RQ-1: What are the key factors that influence the usability of open source software projects from the perspective of users?

RQ-2: Can we identify the key usability factors affecting open source software projects from the viewpoint of OSS developers?

RQ-3: From the perspective of OSS contributors, what are the key factors that affect OSS usability?

RQ-4: What are the key factors that contribute towards the maturity assessment of OSS usability from the perspective of the industry?

RQ-5: Are we able to develop a methodology to assess the usability maturity of an open source software project?

RQ-6: Can specific scales represent the extent of usability maturity in an OSS project?

RQ-7: Can we develop a methodology for evaluating the project's usability maturity level once the assessment results are manifested?

Thus, the answers to the above questions will hopefully provide a comprehensive methodology for performing the usability maturity assessment of OSS projects. In particular, the answers to each individual question will lead to a combination and integration of answers that will solve the overall research problem. For example, the answers to the first four questions provide us with a measuring instrument for assessing the usability maturity of OSS projects. Furthermore, the answers to RQ-5 and RQ-6 foresee the development of the usability maturity model and its related scales, while RQ-7 will provide the rating methodology.

1.3 Problem Definition

This section proposes a usability assessment framework to develop a comprehensive strategy for an OSS project maturity model. Specifically, the framework identifies various maturity scales for the individual dimensions of users, developers, contributors and the industry. The important steps of the research include:

- **Problem Identification – I:** Identification of the factors that contribute directly or indirectly to the maturity assessment of OSS usability

  - **Problem Identification – I-a:** Empirical investigation of the key factors from the perspective of users
  - **Problem Identification – I-b:** Empirical investigation of the key factors from the perspective of developers
  - **Problem Identification – I-c:** Empirical investigation of the key factors from the perspective of contributors
  - **Problem Identification – I-d:** Empirical investigation of the key factors from the industrial perspective
- **Problem Identification – II:** Development of a "Usability Maturity Model" for open source software projects: OS-UMM
  - **Problem Identification – II-a:** Development of maturity scales for assessing OSS usability
  - **Problem Identification – II-b:** Assessment Questionnaires for all usability factors in each maturity scale of OS-UMM
  - **Problem Identification – II-c:** Development of rating methodology for OS-UMM
- **Problem Identification – III:** Application of Usability Maturity Model for assessing the current maturity of an OSS project
  - **Problem Identification – III-a:** Definition of Assessment Methodology
  - **Problem Identification – III-b:** Case Studies
  - **Problem Identification – III-c:** Collection and Interpretation of Assessment Data

1.4 Research Methodology & Approach

Our main objective is to develop a Usability Maturity Model for the assessment of an OSS project. As demonstrated below, Figures 1.1, 1.2 and 1.3 illustrate our research plan to develop the Usability Maturity Model for an OSS project; each stage defines the research objectives and the milestones. We plan to address the three problems identified in Section 1.3 by providing appropriate solutions. Our research methodology includes a comprehensive literature review for defining the maturity level scales based on the key factors of OSS usability and the assessment approach to derive the scales intuitively.

Figure 1.1 illustrates Phase I of our research plan for addressing the solution to problems I, I-a, I-b, I-c and I-d, which are identified in Section 1.3.

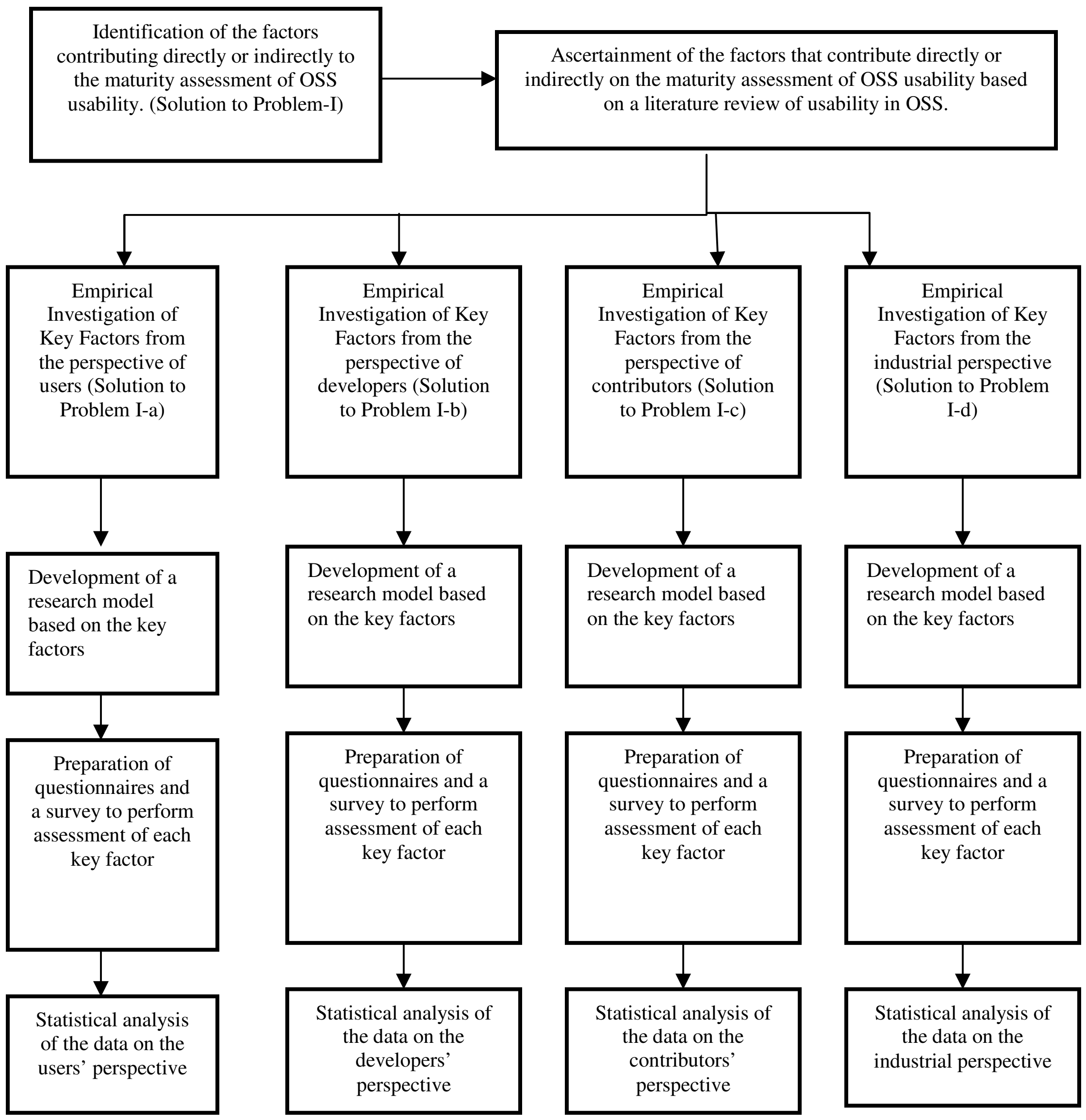

Identification of the factors contributing directly or indirectly to the maturity assessment of OSS usability. (Solution to Problem-I)
Ascertainment of the factors that contribute directly or indirectly on the maturity assessment of OSS usability based on a literature review of usability in OSS.
Empirical Investigation of Key Factors from the perspective of users (Solution to Problem I-a)
Empirical Investigation of Key Factors from the perspective of developers (Solution to Problem I-b)
Empirical Investigation of Key Factors from the perspective of contributors (Solution to Problem I-c)
Empirical Investigation of Key Factors from the industrial perspective (Solution to Problem I-d)
Development of a research model based on the key factors
Development of a research model based on the key factors
Development of a research model based on the key factors
Development of a research model based on the key factors
Preparation of questionnaires and a survey to perform assessment of each key factor
Preparation of questionnaires and a survey to perform assessment of each key factor
Preparation of questionnaires and a survey to perform assessment of each key factor
Preparation of questionnaires and a survey to perform assessment of each key factor
Statistical analysis of the data on the users' perspective
Statistical analysis of the data on the developers' perspective
Statistical analysis of the data on the contributors' perspective
Statistical analysis of the data on the industrial perspective

Figure 1.1: OSS Usability Maturity Model - Research Phase I

Figure 1.2 illustrates Phase II of our research plan for addressing the solution to Problems II, II-a, II-b and II-c, as identified in Section 1.3.

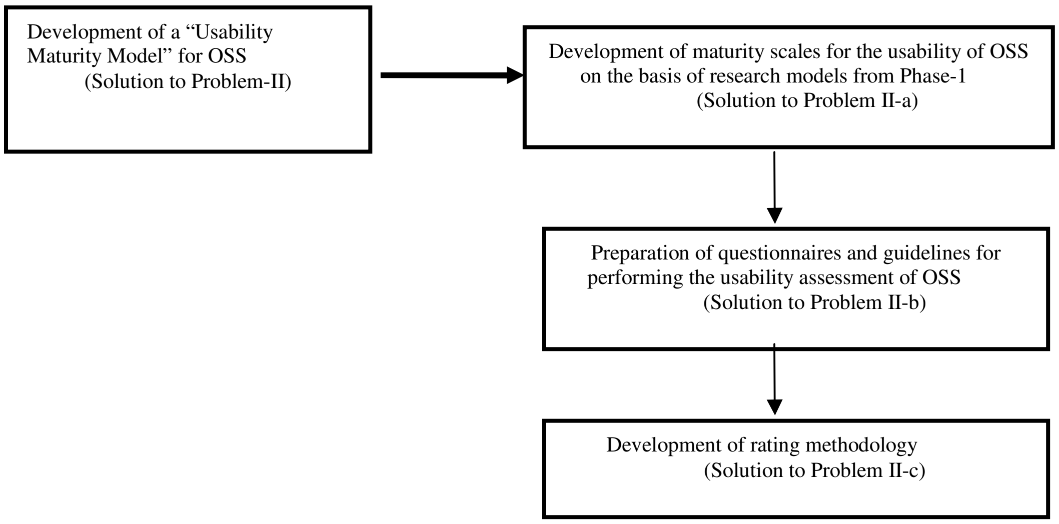

Development of a "Usability Maturity Model" for OSS (Solution to Problem-II)
Development of maturity scales for the usability of OSS on the basis of research models from Phase-1 (Solution to Problem II-a)
Preparation of questionnaires and guidelines for performing the usability assessment of OSS (Solution to Problem II-b)
Development of rating methodology (Solution to Problem II-c)

Figure 1.2: OSS Usability Maturity Model - Research Phase II

Figure 1.3 illustrates Phase III of our research plan for addressing the solution to Problems III, III-a, III-b and III-c, as identified in Section 1.3.

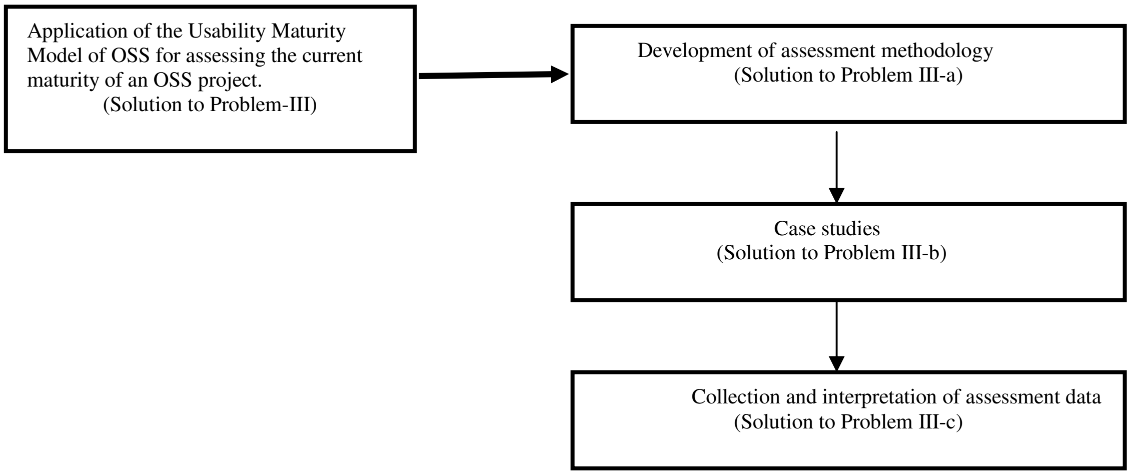

Application of the Usability Maturity Model of OSS for assessing the current maturity of an OSS project. (Solution to Problem-III)
Development of assessment methodology (Solution to Problem III-a)
Case studies (Solution to Problem III-b)
Collection and interpretation of assessment data (Solution to Problem III-c)

Figure 1.3: OSS Usability Maturity Model - Research Phase III

1.5 Contribution of the Thesis

The assessment of open source usability is an important area of research, and a lot of work needs to be done in order to establish a comprehensive and unified strategy for evaluating the maturity of usability in an OSS. Accordingly, this research contributes towards OSS usability assessment in the following ways:

1. An identification of the factors contributing directly or indirectly to the maturity assessment of OSS usability.

2. Empirical evaluation and validation of key factors from the perspective of different stakeholders, including OSS users, developers, contributors and the industry.

3. The development of an open source usability maturity model using the empirically validated key factors as a measuring instrument.

4. The application of OS-UMM for assessing the current maturity of an OSS project.

1.6 Overview of the Thesis

The organization of this thesis is based on five articles that have been published or are currently under review in software engineering journals. These papers are cited accordingly in the corresponding chapters. In Chapter 2, we present the definition of OSS and distribution terms as well as provide a literature review that is related to OSS quality issues. Additionally, we discuss general issues related to usability as well as a literature review of usability assessment and measurement. Lastly, we examine usability issues in open source projects and provide a brief review of existing usability and maturity assessment models that motivated this research.

Chapter 3 presents an empirical investigation for studying the impact of key factors on OSS usability from the perspective of end users, establishing the relationship between the key usability factors from the viewpoint of end users and OSS usability. The study conducted and reported in this chapter can enhance the understanding of OSS designers and developers, especially in terms of the relationships between the key usability factors and their projects. Consequently, the OSS development community can use this research to address usability issues in their projects. Therefore, this empirical investigation provides justification for considering these key factors as a measuring instrument for a maturity model that assesses the usability of open source software projects.

OSS developers need to consider several key usability factors in order to improve the general usability aspects of software as well as their specific projects. Accordingly, Chapter 4 presents a research model of an empirical investigation that establishes a relationship between the key usability factors from the perspective of developers and OSS usability. This study is among a series of four studies that are conducted for the improvement of OSS usability from the viewpoints of OSS developers, users, contributors and the industry. The results of this investigation show that the key factors in our research model assist in improving OSS usability.

Chapter 5 presents an empirical investigation for studying the impact of several key factors on OSS usability from the perspective of contributors, including architects, designers, developers, testers and users. In this study, we investigate the effects of key usability factors on OSS usability and discover answer to the research question stated in this investigation. The study shall enable OSS development teams to improve their understanding of the relationship between the key factors and the usability of their projects.

Although user-centered designs are gaining popularity in OSS, many design scenarios still do not consider usability as one of their primary goals. Accordingly, Chapter 6 presents a research model of the study establishing a relationship between the key usability factors and OSS usability from the industrial perspective. This research study is the fourth in the series of our empirical investigations, in which we analyze the impact of the key usability factors, including understandability, learnability, operability and attractiveness, on OSS usability. Indeed, these four key factors have been incorporated from the standard ISO/IEC 9126-1 (2001). The empirical results of our study strongly support the hypotheses that understandability, learnability, operability and attractiveness have a positive impact on the usability of OSS projects.

Chapter 7 presents a usability maturity model specifically aimed at usability-related issues for open source projects. In particular, the model examines the relationship between open source projects and their usability aspects. The measuring instrument of the model contains factors that have been selected from four of our empirical studies, which examine the perspectives of OSS users, developers, contributors and the industry. In addition to presenting the usability maturity model, this chapter discusses assessment questionnaires, a rating methodology and two case studies.

Finally, Chapter 8 concludes this thesis and highlights the summary of the research contributions of this dissertation in the area of open source software usability. This chapter also discusses the future research directions in the OSS usability assessment.

Chapter 2

2 Open Source Software Usability Assessment

The use of free and open source software has increased significantly in recent years, largely due to the accessibility and availability of the Internet. Although it is generally believed that OSS is used mostly by technically adept users, obscuring the boundary between the developers and users, most users of OSS are actually novice or non-technical users. Thus, three challenges for improving OSS usability include gaining a better understanding of users' expectations through empirical investigation, adapting new approaches to OSS designs, and quantifying usability metrics.

2.1 OSS – Definition and Distribution Terms

Open source refers to the use of shared source codes, open standards, and collaboration among software developers and users worldwide for building software, identifying and correct errors, and making enhancements (O'Reilly 1999). According to Hansen et al. (2002), OSS means that *"the source code is distributed along with the executable program. It is free to use. It includes a license allowing anyone to modify and redistribute the software."* Open Source Initiative (OSI) is a non-profitable corporation with the aim of enhancing awareness about OSS and its benefits. According to OSI, OSS is not only about free access to the source code. Rather, a specific set of criteria, provided in Table 2.1, must be met in order to fulfill the distribution terms of open-source software.

Table 2.1: Open Source Distribution Terms (opensource.org/docs/osd)

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.
7. Distribution of License The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
8. License Must Not Be Specific to a Product The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.
9. License Must Not Restrict Other Software The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.
10. License Must Be Technology-Neutral No provision of the license may be predicated on any individual technology or style of interface.

2.2 OSS - Quality Issues

The collaborative nature of the OSS culture utilizes a volunteer community that conducts its development activities in a decentralized environment, effectively lowering production costs and improving software quality (Raymond 2001). Laplante et al. (2007) observe that many organizations are still reluctant to use open source software, mainly due to *"an inherent distrust of OSS quality."* However, according to these authors, the use of OSS, especially in comparison to closed proprietary software, produces a higher quality of software.

Due to the involvement of and acceptance by big commercial IT vendors, OSS products have progressed from a fringe activity to enter into the mainstream (Fitzgerald 2006). von Krogh and Spaeth (2007) argue that the possession of an open license is one of the major reasons for the existence of OSS. Specifically, they believe that in order to maximize the efficiency of this open source phenomena, *"information systems research should remain open to dialog with other areas and disciplines."* Levesque (2005) argues that OSS must address issues such as user interface design, documentation, feature-centric development, self-programming and *"religious blindness"* in order to be widely accepted.

However, Polancic et al. (2004) question the quality of OSS projects, proposing an assessment model for evaluating these products. Paulson et al. (2004) maintain that defects are found and fixed more rapidly in OSS as compared to closed source software products. In their empirical findings, Capra et al. (2008) suggest that *"quality, mainly measured in terms of coupling and inheritance, is a must in OS projects to enable more open governance mechanisms, which, in turn, increase development effort, while quality per se does not increase effort."* Koch and Neumann (2008) explore the influence of different forms of OSS development processes on the resulting software and verify their effects on different quality aspects. These authors attempt to ascertain whether different variants of OSS development processes significantly influence the resulting products, analyzing *"the impact on quality and design associated with the numbers of contributors and the amount of their work, using the GINI coefficient as a measure of inequality within the developer group."*

Zhao and Elbaum (2003) investigate the relationship between the unique features of OSS and software quality. In particular, they examine how software quality assurance is performed in the OSS environment, how is it different from traditional software development models, and whether or not these differences are advantageous in creating better quality OSS products. The outcome of their study demonstrates that a new dimension in large-scale distributed software development has occurred as a result of the OSS development approach. By developing an OSS success model from an existing Information Systems (IS) model, Lee et al. (2009) identify the determinants for OSS success and conclude that “*software quality and community service quality have significant effects on user satisfaction.”* Furthermore, they recommend that *“usefulness, ease of use, and reliability”* are some of the major factors that OSS practitioners should heed in order to improve OSS quality.

Although OSS products are ultimately dependent on the skills of the developers, Hedberg et al. (2007) believe that high-quality software can be produced by OSS. Similarly, Michlmayr et al. (2005), argue that to some extent, OSS applications meet the expectations and quality requirements of users. However, these authors emphasize the need for empirical investigations to study quality related problems in OSS. Specifically, they argue that OSS *“is an area where synergies between the academic world and the free software community may be possible. The implementation of quality improvement strategies also requires tools and metrics to measure and assess different aspects of quality.”*

We can thus conclude here that although there are many examples of high quality OSS, quality-related issues need to be addressed at a high priority.

2.3 Usability - General Issues, Assessment and Measurement

In the ISO 9241-11 (1997) Standard, usability is defined as *“the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.”* However, ISO/IEC 9126-1 (2001), states that usability is *“the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions.”*

Seffah and Metzker (2004) highlight some obstacles of usability in software engineering (SE). For instance, software development organizations generally struggle to understand the user-centered design techniques that are developed by HCI teams, as developers often have specific usability techniques that they have created. According to these authors, these two communities could achieve a harmonious coexistence if these obstacles could be realized and hence avoided. In order to understand and overcome such obstacles, a forum should be created to share ideas between the two fields so that a cohesive relationship can be built between them.

Te'eni (2007) maintains that for thirty years, organizational tasks and contexts have been the main focus of management information systems. Consequently, research should be conducted to determine which higher level tasks users wish to perform and how users interact with the computer to achieve these tasks. Koppelman and Dijk (2006) focus on the role of clients and users in projects. Specifically, they discuss issues such as learning to deal with different stakeholders who view the product from different perspectives, communicating with these various groups and involving the users and clients in the design process. In their examination, they suggest that the designers should not simply rely on their own experiences and instincts.

Aspects of usability cannot be improved unless there are ways to test and measure them. While studying current practices in usability measurement, Hornbaek (2006) acknowledges the fact that usability cannot be directly measured. He identifies several challenges to usability measurement, such as understanding the relationship between objective and subjective measures of usability, measuring learnability, extending satisfaction measures beyond post-use questionnaires and studying correlations between different measures. In discussing two usability case studies, Craven and Booth (2006) state that *"cognitive walkthroughs, heuristic evaluation, expert usability evaluations and usability audits, are often directed more towards expert usability testing rather than managing the user testing in-house."* Miller (2006) considers usability testing as a process that starts with a project's beginning and continues through prototyping, completion, and even after releasing. He maintains that *"just because the project is done doesn't mean the project's usability work is finished."* Moreover, he believes that in such studies, open-ended questions should be asked to the participants. Since he considers usability as a quality assurance aspect, he does not see any incompatibility between usability testing and rational product processes. Çetin and Göktürk (2008) observe that although software testing traditionally consumes considerable time, there is a limited amount of formal testing conducted by OSS developers. Furthermore, the authors realize that usability is a non-functional quality attribute, and, as it is a *"subjective"* matter, it cannot be measured directly. Lastly, they maintain that OSS developers need to recognize the usability level of their projects.

Golden et al. (2005) support the idea of addressing usability issues at the software architecture design level. They insist that the separation of usability concerns from functionality at the architecture level is ineffective, even when the intention is to consider usability issues at a later testing stage. Rather, this approach leads to extensive re-structuring and even the *"re-architecting of software systems."*

2.4 Usability Issues in Open Source Projects

In the future, OSS is expected to have an increasing diversity of users, including those with technical and non-technical backgrounds as well as those from varying cultures, each with their own needs, expectations and demands. Even in the environment of closed proprietary software, usability is a complicated issue; however, in OSS, it is even more difficult, especially considering that the domain is relatively newer with developers working on a voluntarily basis and where there is generally no HCI or usability expert.

While examining usability practices in OSS, Nichols and Twidale (2006) highlight the need to address usability issues more arduously. These authors discuss the failure of certain commercial closed source software projects, which result from unusable systems or poor handling usability issues. Specifically, they believe that these failures are an indication of usability being an *"unresolved"* issue even in proprietary software, which is considered more usable than OSS, as closed source software is more mature than OSS, and it is equipped with more resources, both in terms of experienced manpower and financial resources. According to the authors, *"research in open source usability has the potential to be valuable to all kinds of software development, not just OSS."* In their research, they focus on finding ways to ease usability bug reporting and involve usability experts during the software analysis and design phases.

Aberdour (2007) contrasts the "*formal and structured testing*" that is typical in closed software development with the "*unstructured and informal testing*" in OSS development. On the other hand, Hedberg et al. (2007) maintain that the processes of "*test coverage, test-driven development and testing performed by developers*" require more attention in OSS projects through formal and detailed test plans that ensure errors are caught before the release of the software. Andreasen et al. (2007) argue that remote usability testing is very relevant in an OSS environment. Specifically, they believe that remote usability testing has the potential to cross geographical and organizational boundaries for supporting OSS development.

Viorres et al. (2007) highlight a general trend in OSS development; instead of following the software engineering practice of design, specifications, testing and prototyping, most OSS systems incorporate a *"bottom-up"* approach that focuses on the development of technical issues and individual components. In this approach, system modeling, as well as the user interface and related issues, are a lower priority. Otte et al. (2008) discuss the structured defect handling processes, where there is a significant use of configuration and bug tracking tools. Specifically, they highlight the high rate of user participation, user testing and peer reviews in OSS projects.

Çetin and Göktürk (2007) believe that traditionally, usability and user-centric designs have been challenging in open source software. Nevertheless, they indicate their optimistic belief that usability is beginning to become institutionalized in open source projects. They also refer to lack of user-centered design in an OSS development environment, arguing that that there is a need for more collaboration between interaction designers and OSS developers as well as contributions from usability expert in order to improve the overall quality of OSS.

According to Feller and Fitzgerald (2000), OSS users have traditionally been the developers, testers, documenters or field experts, resulting in an overlap between developers and users. However, since OSS has entered into the mainstream and received support from big corporations such as IBM, Apple and Oracle, novice users have become interested in OSS. Zhao and Deek (2005) identify a weak relationship between OSS and usability. Specifically, they maintain that the requests of experienced OSS users do not necessarily reflect the requirements of non-experienced users. These authors emphasize the importance of building effective communication systems for involving novice users and endowing them with usability knowledge. Similarly, Bodker et al. (2007) perceive the lack of user-friendly products in OSS as a serious threat towards its popularity, arguing that this paucity results because OSS developers do not have a complete understanding of user situations. In their empirical study, Andreasen et al. (2006) found that although OSS developers realize the importance of end users, they still fail to prioritize usability related issues. Specifically, these authors state that *"currently, most developers have a very limited understanding of usability. Moreover, there is a lack of resources and evaluation methods fitting into the OSS paradigm."* Similarly, Yunwen and Kishida (2003) stress the need for more interaction between software users and developers, suggesting that OSS projects can succeed with the development of a collaborative platform between the two communities. These authors realize that OSS differs from closed proprietary software in more ways than the user having free access and the right to modify a source code. Rather *"the fundamental difference is role transformation of the people involved in the project."* Unlike closed source software projects, the OSS development environment does not clearly and strictly define the boundaries between users and developers.

Porter et al. (2006) observe the inconsistency in open source software, which mainly results from "*short feedback loops between users and core developers."* They believe that the frequent release of beta versions frustrates many users who would prefer more stable software. They also identify unsystematic and undocumented software testing in OSS. In their empirical study, Stamelos et al. (2002) conclude that user satisfaction is related to the average size of components in an open source application. In addition, they support the idea of OSS having its own quality standards. Similarly, Crowston et al. (2003) emphasize that the measurement of success and quality of an OSS project is necessary, as millions of users are dependent on OSS systems. In particular, they recommend the collection of user feedback by building a survey into the software. They also list a number of factors that indicate the success and popularity of an OSS project;

these factors include the number of users, the quality of codes and documentation, the user ratings, the downloads, and the reuse of code.

Hedberg et al. (2007) consider usability as a research area in OSS that needs to be examined. Specifically, they state that *"user feedback should be sought early, and the design solution should be iterated based on the user feedback."* Moreover, they associate the improvement in OSS quality with usability and with issues related to *"naïve, non computer professional"* users, who, according to these authors, are increasing on a daily basis, and hence, their expectations are also increasing. These authors observe that since software developers are neither paid nor have any formal authority, the only means of assuring OSS quality is their commitment. They also highlight a few problem areas, such as the obscure distinction between users and developers, the lack of usability knowledge in typical OSS developers and the absence of HCI personnel or usability experts in the development of an OSS project.

2.5 Existing Usability Models

A literature survey of existing work in this research area reveals many usability models. However, the vast majority of these models do not provide validation for their proposed metrics or they are not specifically aimed at issues related to open source projects.

2.5.1 A Comprehensive Model of Usability by Winter et al. (2008)

Winter et al. (2008) propose a two-dimensional usability model relating system properties to user activities. The authors claim that their model *"fosters preciseness and completeness", "serves as a central and structured knowledge base for the entire quality assurance*," provides a basis for analysis and measurement and describes the system's usability in relation to the activities performed by the user. Furthermore, they assert that their usability model provides a suitable basis for domain-specific or company-specific models. The relationship between the system property and the activity decompositions as well as their interrelations, as visualized by the authors, is depicted in Figure 2.1.

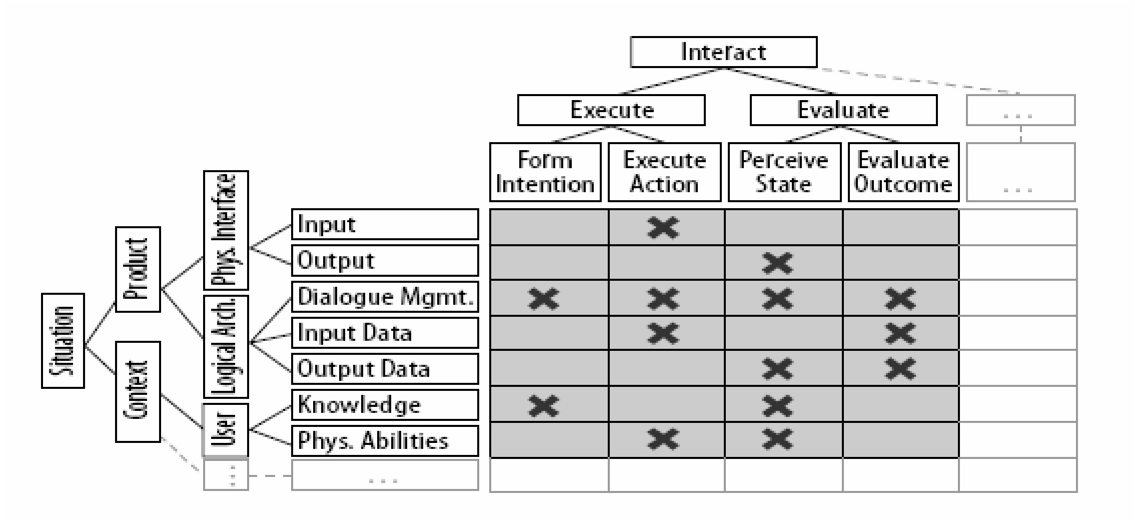


Figure 2.1: A Comprehensive Model of Usability (Winter et al. 2008)

This model separates system properties and activities, thus specifically describing quality attributes and their related impact on the use of software. Furthermore, the model distinguishes between facts and attributes; facts are a way of hierarchically depicting the system use without quality criteria, whereas attributes provide the facts with low-level quality criteria, such as consistency, ambiguousness or existence. Specifically, the model hierarchically decomposes user activities as part of the interaction with the system. Based on ISO 9126-4, the usability quality model makes use of *"Efficiency"* as -the utilization of resources, *"Effectiveness"* as the sharing of successful tasks, *"Satisfaction"* as the enjoyment of product use and *"Safety"* for defining the usability goals of the system. The root node *"use"* shown in the activity tree of the usability model denotes any use of the software-based system. The two offspring of the root node are *"execution of secondary tasks"* and *"interacting with the product."* The *"execution of secondary tasks"* deals with all additional user tasks that are not directly related to the software product, and the activity *"interacting with the product"* is further decomposed into seven stages of action. A case study has been presented in order to evaluate this usability model using the ISO 15005 Standard.

However, the model requires validation as well as refinement to more specific contexts. The authors state that *"other quality attributes, e.g. reliability, will also be modeled by means of the meta-model to investigate whether this approach works for all attributes."*

2.5.2 A Measurement Based Framework for Assessment of Usability-Centricness of Open Source Software Projects by Çetin and Göktürk (2008)

Çetin and Göktürk (2008) maintain that OSS usability aspects cannot be improved unless there are ways to quantitatively measure them. Specifically, they stress the importance of testing and measurement by stating that "*one can't improve what is not measured.*" Additionally, the authors highlight the OSS development approach, which entails developing *"feature-centric projects rather than following a user centered design."* According to them, OSS developers generally do not realize the importance of usability. As a result, they state that *"usability can also be used as a set of approaches while creating products, leading to reducing costs and increase the user satisfaction, by including people in the design stage who are directly the users of that particular product."* These authors believe that OSS lacks an understanding of inner principles as well as an in-depth analysis of metrics and quality work. Accordingly, they have conducted two surveys to examine the correlation between the working procedures of usability and OSS developers.

Çetin and Göktürk (2008) define the user-centered design process as *"a design philosophy in which the requirements of the user of a product are given key attention during the whole project life cycle."* Based on their survey results, they conclude that the *"user centered design process should be a part of a project which is carried online."* Furthermore, their survey results, as well as their literature review, highlight the need for a bug tracker, especially one with a proper configuration and user-friendly features. In addition, they believe that for an OSS project, *"a user profile, a proper guideline and critical user tasks list and a conceptual modeling is also required."* Moreover, their survey results consider the services of usability experts as vital for OSS projects. Finally, the authors believe that *"designers need to learn how different users from distinct cultures interact with interfaces, and benefit from cultural interaction models in order to build culture-friendly interfaces. For an open source project, this requires a methodology and a list of guidelines to aid developers in building cultural interfaces."*

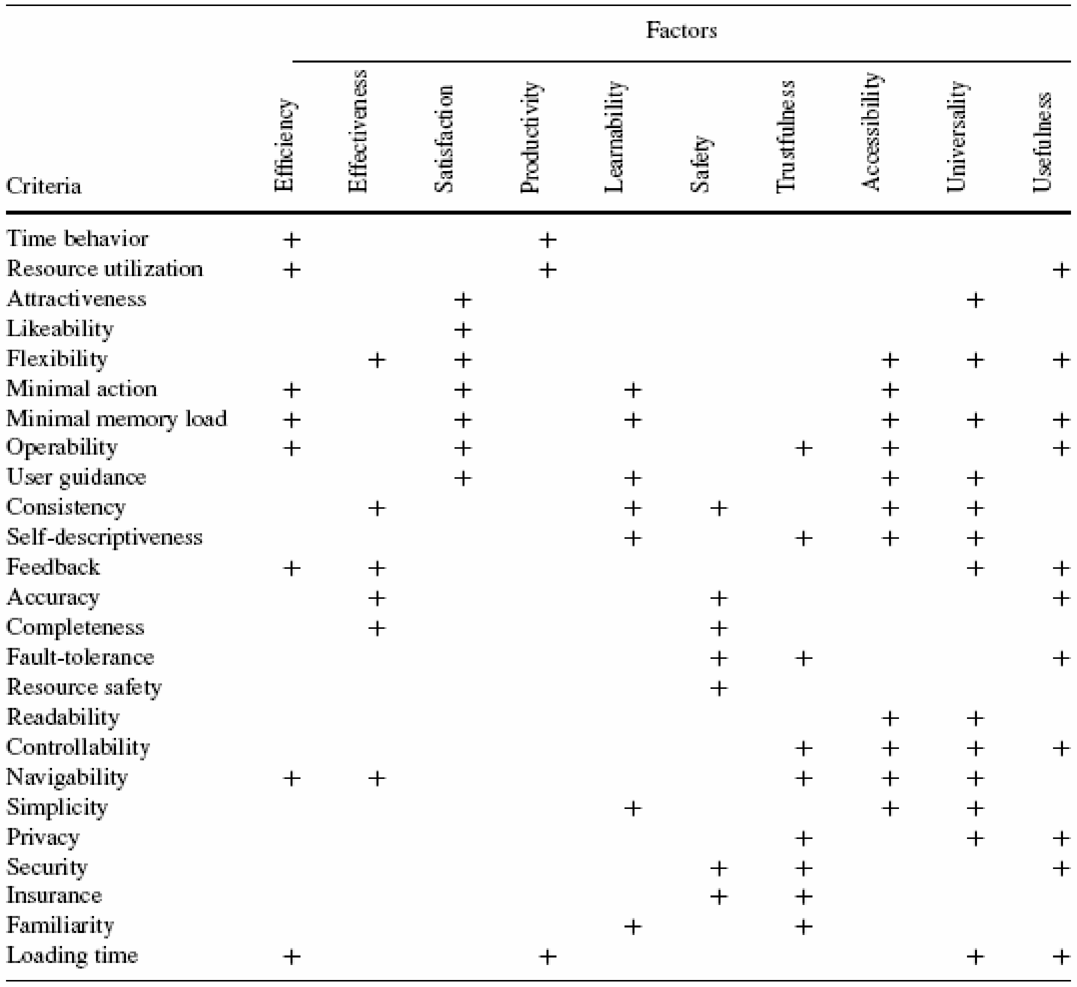
Accordingly, the authors have proposed a metric model using literature research and survey findings to measure and analyze the usability of an OSS project. However, no validation of the proposed metrics has been presented, as they state that *"the main lacking part of this paper is that no validation of the proposed metrics has been done. It's required to apply the provided metrics to various F/OSS projects for which their usability is known to a specific extent. Another method could be to measure the usability of these applications and try to find a correlation between them."*

2.5.3 Quality in Use Integrated Measurement (QUIM) Model by Seffah et al. (2006)

Seffah et al. (2006) review and highlight the benefits and shortcomings of existing standards and models, and subsequently, they present a consolidated, hierarchical Quality in Use Integrated Measurement (QUIM) model for measuring software usability. Specifically, they believe that their model *"brings together usability factors, criteria, metrics, and data mentioned in various standards or models for software quality and defines them and their relations with one another in a consistent way."*

This hierarchical model consists of 10 factors representing a specific facet of usability; these factors are decomposed into a total of 26 sub-factors, which are further divided into 127 specific metrics. The 10 factors include Efficiency, Effectiveness, Productivity, Satisfaction, Learnability, Safety, Trustfulness, Accessibility, Universality, and Usefulness. Based on these factors, the 26 sub-factors are quantifiable by means of one or more specific metrics, which consist of either a formula or countable data. The relationship between the usability factors and their sub-factors is depicted in Table 2.2. Additionally, the authors have presented the QUIM Editor as a usability measurement toolbox for supporting the activities described in the model.

Table 2.2: Relations between factors and criteria in QUIM (Seffah et al. 2006)



Criteria	Factors									
	Efficiency	Effectiveness	Satisfaction	Productivity	Learnability	Safety	Trustfulness	Accessibility	Universality	Usefulness
Time behavior	+			+						
Resource utilization	+			+						+
Attractiveness			+						+	
Likeability			+							
Flexibility		+	+					+	+	+
Minimal action	+		+		+			+		
Minimal memory load	+		+		+			+	+	+
Operability	+		+				+	+		+
User guidance			+		+			+	+	
Consistency		+			+	+		+	+	
Self-descriptiveness					+		+	+	+	
Feedback	+	+							+	+
Accuracy		+				+				+
Completeness		+				+				
Fault-tolerance						+	+			+
Resource safety						+				
Readability								+	+	
Controllability							+	+	+	+
Navigability	+	+					+	+	+	
Simplicity					+			+	+	
Privacy							+		+	+
Security						+	+			+
Insurance						+	+			
Familiarity					+		+			
Loading time	+			+					+	+

However, the authors consider their model as a first step towards their goal, stating that *"more attention to the information about the user and the context is also required to facilitate the selection and the customization of many of the proposed metrics and higher-level criteria."*

2.5.4 A Consolidated Model for the Evaluation of Software Usability by Abran et al. (2003)

Abran et al. (2003) highlight the importance of usability, maintaining that it is a required quality attribute for future software development. However, they consider a software design with strong usability as a significant challenge, particularly if there is no direct access to the end users. Based on an analysis of ISO-9241-11 and ISO- 9126, the authors present a consolidated usability model. Consequently, they discover that ISO-9241-11 was developed by HCI specialists, whereas ISO-9126 resulted from the efforts of software engineering experts. Subsequently, the authors propose a revised integrated usability model that includes *"learnability"* and *"security"* as their baseline from ISO-9241. The structure of the model is shown in Figure 2.2.

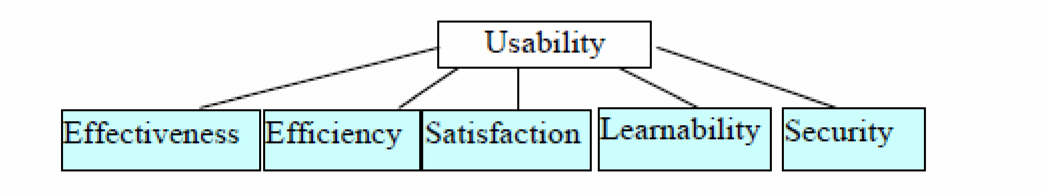


Figure 2.2: Revised Usability Model (Abran et al. 2003)

Specific measures of these characteristics have also been presented by using the structured hierarchy from ISO-9126. However, the authors realize that more work needs to be done in the area of usability modeling; specifically, this area requires a greater level of consensus among researchers, and it needs a more comprehensive model.

2.5.5 The Usability Engineering Process Model (UEPM) by Granollers et al. (2003)

Granollers et al. (2003) have created The Usability Engineering Process Model (UEPM), which integrates SE with HCI activities and focuses on usability. The structure of the model is shown in Figure 2.3.

The UEPM contains three columns: the traditional Software Engineering life cycle, Prototyping, and Evaluation. SE protection activities, such as *Configuration Management, Software Quality Assurance and Risk Management,* are also used for supporting the software development process. Specifically, the authors present their *"Process Model as an offer for the development of interactive applications integrating the specific models and tasks of usability with the life cycle of SE."*

As the authors consider the validation of the model by specific projects as vital, they state that they are in the process of developing different paradigm applications.

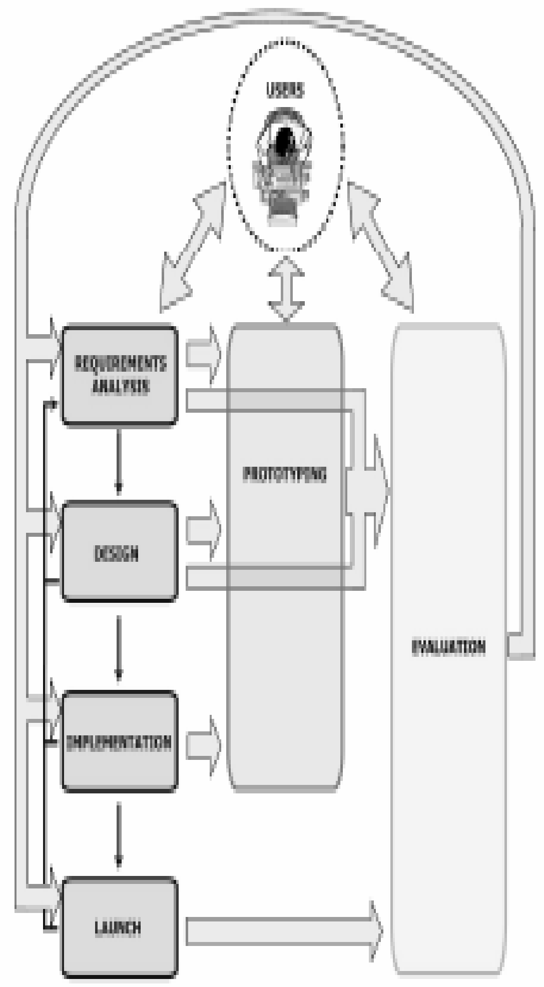


Figure 2.3: Usability Engineering Process Model – UEPM (Granollers et al. 2003)

2.6 Software Process Assessment Models

A software process consists of a set of tools, methods and practices used to produce a software product (Humphrey 1990). Software process assessment determines the current status of an organization's process and identifies the areas that need improvement. Within this assessment, *"a maturity level is a defined evolutionary plateau for organizational process improvement."* (Chrissis et. al. 2006).

2.6.1 Capability Maturity Model (CMM)

The Capability Maturity Model (CMM) proposed by the Software Engineering Institute (SEI) has been accepted by the software industry. The structure of CMM is shown in Figure 2.4; each maturity level is composed of several Key Process Areas (KPAs), and each KPA is organized into common features that specify the essential practices that, when collectively addressed, accomplish the goals of the KPA (Paulk et. al. 1995).

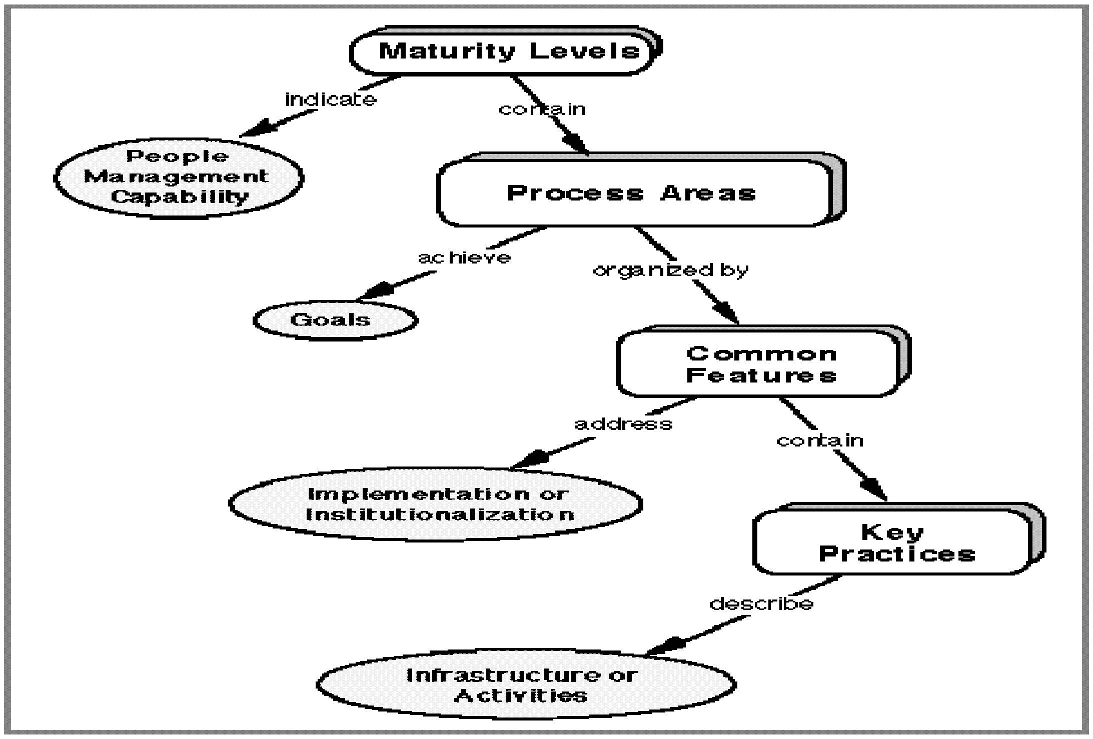


Figure 2.4: Capability Maturity Model – CMM® Structure (Paulk et. al. 1995)

The CMM is divided into five maturity levels, ranging from Level 1 to Level 5, as shown in Figure 2.5. Level 1 is called *"Initial"*, a level at which software processes are characterized as *"ad hoc"*, and it is assumed that at this level, the organization operates without formal procedures or project plans. In an organization at the *"repeatable"* level, Level 2, project controls have been established over quality assurance, change control, cost and schedule. At Level 3, referred to as the *"defined"* level, the management and engineering activities are well-documented, standardized, and integrated across the organization. At the *"managed"* level, Level 4, measures of software process and product quality are collected so that process effectiveness can be determined in a quantitative manner. For an organization to achieve the fifth or *"optimizing"* level, they require quantitative feedback in order to achieve continuous process improvement.

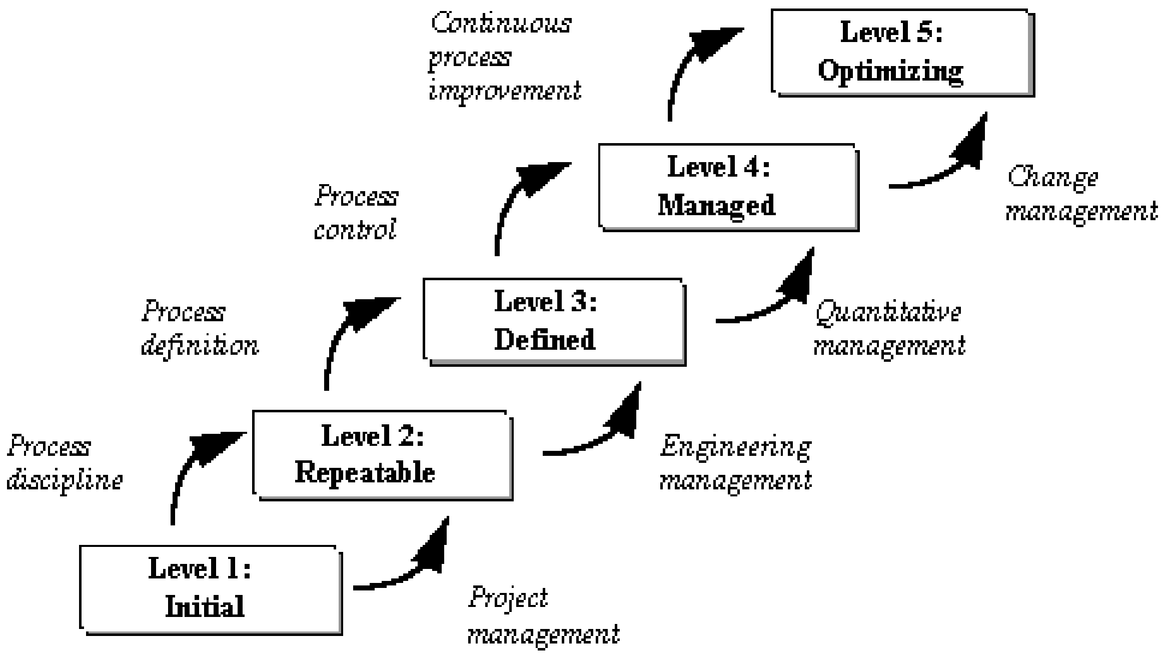


Figure 2.5: CMM Maturity Framework: Five Levels (Paulk et. al. 1995)

2.6.2 Software Process Improvement and Capability dEtermination (SPICE)

"SPICE provides a measure of the maturity of a process against a defined model of best practice, rather than a pass/fail result as with ISO9001" (Mackie 1997). Software Process Improvement and Capability dEtermination (SPICE), or ISO 15504, defines a reference model of process capabilities, thus forming the basis of any model to be used and providing a framework for software process assessment. Additionally, it extends the five maturity models proposed by CMM, to six, as shown in Table 2.3.

2.6.3 Capability Maturity Model Integration (CMMI)

Over time, CMM has evolved into the Capability Maturity Model Integration (CMMI) (Chrissis et. al. 2006), which results from the evolution of The Capability Maturity Model for Software (SW-CMM), The Systems Engineering Capability Model (SECM) and The Integrated Product Development Capability Maturity Model (IPD-CMM). Two improvement paths are supported by CMMI; the term *"capability level"* denotes continuous representation, and the term *"maturity level"* indicates staged representation. According to Chrissis et. al. (2006), *"continuous representation is concerned with selecting both a particular process area to improve and the desired capability level for that process area."* Because it is important to indicate whether a process is incomplete or performed, the continuous representation path has six capability levels, with the *"incomplete"* level as Level 0. In contrast, staged representation, which is concerned with qualitatively illustrating the maturity of an organization, has five maturity levels, as shown in Table 2.3.

2.6.4 Comparison of CMM, SPICE and CMMI

A comparison of scales used in CMM, SPICE and CMMI is presented in Table 2.3.

Table 2.3: Capability Maturity Levels in CMM, SPICE and CMMI

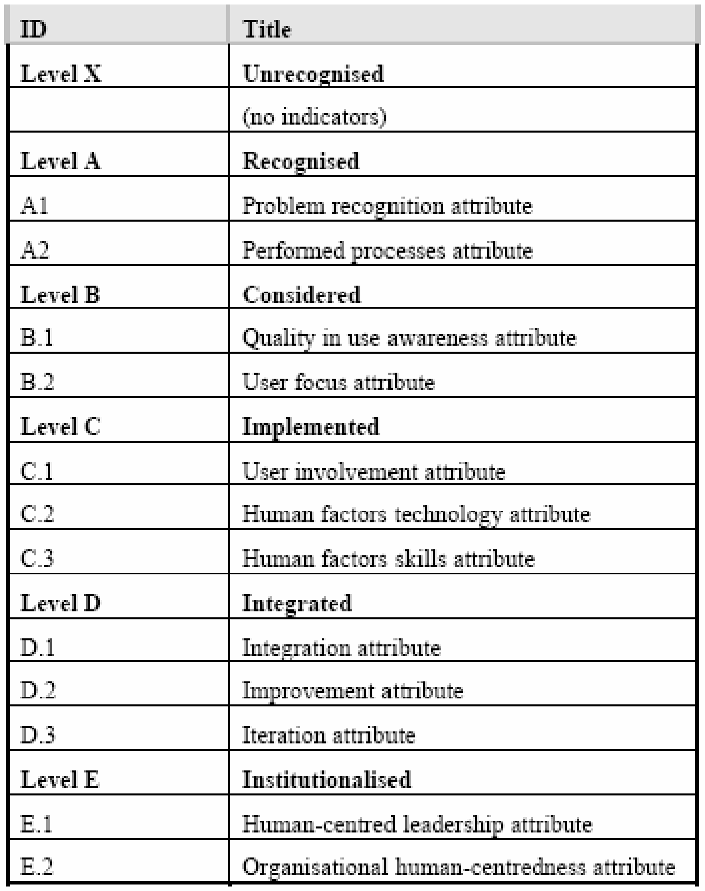
Level	**CMM** (Paulk et. al. 1995)	**SPICE** (Mackie 1997)	**CMMI** Staged Representation (Chrissis et. al. 2006)	**CMMI** Continuous Representation (Chrissis et. al. 2006)
Level 0	N/A	Incomplete	N/A	Incomplete
Level 1	Initial	Performed	Initial	Performed
Level 2	Repeatable	Managed	Managed	Managed
Level 3	Defined	Established	Defined	Defined
Level 4	Managed	Predictable	Quantitatively Managed	Quantitatively Managed
Level 5	Optimizing	Optimising	Optimizing	Optimizing

2.7 Existing Usability Maturity Models

2.7.1 Usability Maturity Model: Human Centredness Scale by Earthy (1998)

Earthy (1998)'s "*Usability Maturity Model: Human Centredness Scale"* demonstrates the way in which organizations can progress through six levels of human-centered processes. Accordingly, the model has six levels that are defined by a set of attributes, each of which is described by one or more management practices that apply to that particular level. The Human-Centredness Scale, which conforms to ISO 15504, describes six sets of process, technology and attitude indicators that define levels of organizational maturity relating to human-centredness. Overall, the model is designed to assess the maturity level of a human-centred approach in an organisation, and different management activities are used to define each maturity level, as shown in Table 2.4.

Table 2.4: Maturity Levels and Process Attributes (Earthy 1998)



ID	Title
Level X	**Unrecognised**
	(no indicators)
Level A	**Recognised**
A1	Problem recognition attribute
A2	Performed processes attribute
Level B	**Considered**
B.1	Quality in use awareness attribute
B.2	User focus attribute
Level C	**Implemented**
C.1	User involvement attribute
C.2	Human factors technology attribute
C.3	Human factors skills attribute
Level D	**Integrated**
D.1	Integration attribute
D.2	Improvement attribute
D.3	Iteration attribute
Level E	**Institutionalised**
E.1	Human-centred leadership attribute
E.2	Organisational human-centredness attribute

Moreover, the model discusses the extent to which each attribute needs be comprehended for a particular maturity level to be achieved. According to Earthy, *“the scale is mainly intended for use in a ‘stand-alone’ mode. In this mode the assessor will usually be a human factors consultant or a member of an organisation’s staff charged with determining and improving an organisation’s level of capability in human factors, or with improving the quality in use of an organisation’s products.”* In addition, Earthy discusses an assessment methodology for evaluating the rating of human-centred activities in an organisation.

Despite the thoroughness of the model, however, there is currently no empirical validation of the proposed attributes that determine a capability level achievement in Earthy’s maturity model.

2.7.2 Usability Maturity Model: Processes by Earthy (1999)

In his report, Earthy (1999) presents a “*Usability Maturity Model: Processes*” for assessing an organization’s capability to perform activities that are related to human-centered processes. Shown in Figure 2.6, *“the model conforms to and extends ISO DIS 13407 Human-Centred Design Processes for Interactive Systems.”* Specifically, the model comprises seven sets of practices describing the required accomplishments during the lifecycle, thus representing and including users.

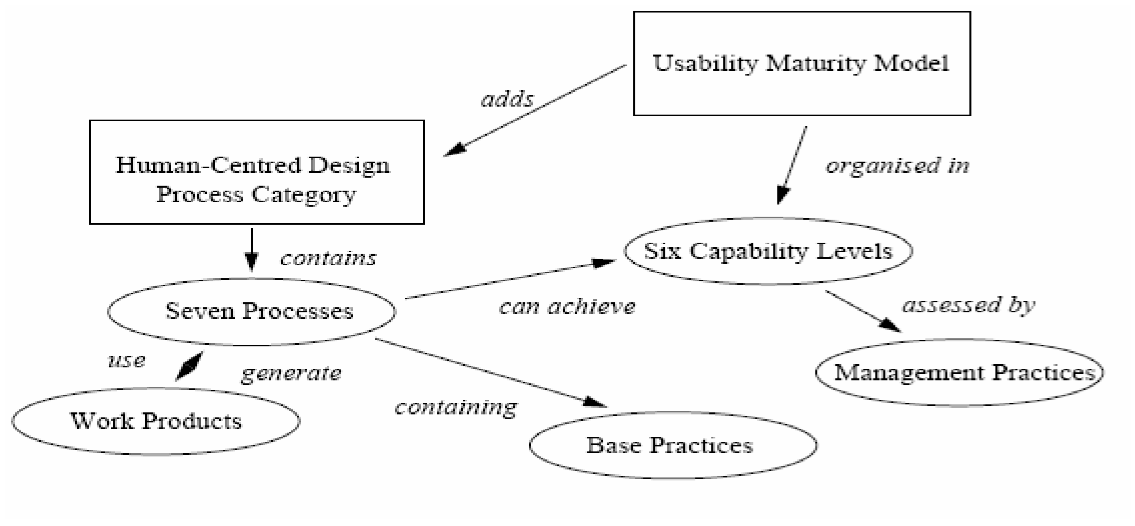


Figure 2.6: Entity relationship diagram of the model (Earthy 1999)

Additionally, the model consists of seven human-centred development (HCD) processes that may occur during system development. The processes, HCD.1 to HCD.7, are named and described as the following: "*Ensure HCD content in systems strategy, Plan and manage the human-centred design process, Specify the user and organisational requirements, Understand and specify the context of use, Produce design solutions, Evaluate designs against requirements, Introduce and operate the system.*" Each of these processes contains different base practices, which help in achieving a certain capability level. Conforming to ISO 15504, this model also entails six capability levels: *"Level 0 Incomplete (not able to carry out process), Level 1 Performed (individuals carry out process), Level 2 Managed (quality, time and resource requirements for process known and controlled), Level 3 Established (process carried out as specified by organisation, resources are defined), Level 4 Predictable (performance of process within predicted resource and quality limits),Level 5 Optimising (organisation can reliably tailor process to particular requirements)."* Each capability level above 0 is associated with certain attributes of the process that demonstrate the achievement of that level.

It is stated that the model *"can be used in the assessment of an organisation's capability to carry out the human-centred processes described in the model."* However, for process assessment, *"the reader is referred to ISO 15504 for details of the qualification of assessors, quality processes associated with assessments etc."* Furthermore, no empirical validation of the proposed attributes, which determine a capability achievement in each level, has been provided.

2.7.3 The UUMM: User and Usability Maturity Model by Lethbridge (2007)

Lethbridge (2007) presents the User and Usability Maturity Model (UUMM), which is based on the Capability Maturity Model. The UUMM, which can assess an organization's capabilities with users and usability-related issues, covers four dimensions, including *"involvement with users, training of the team, development processes and evaluation processes."* The author suggests that the profitability of organizations could be improved with the use of UUMM or similar models. In particular, Lethbridge recommends that *"all* *organizations should strive for level 3 of the UUMM, and those with more than about 5 software professionals ought to profitably be able to reach level 4."*

However, the authors have not yet used the model for conducting an assessment of an organization.

2.7.4 Comparison of Usability Maturity Models

A comparison of the existing usability maturity models and their respective scales is shown in Table 2.5.

Table 2.5: Usability Maturity Models and their Scales

Usability Maturity Model: Human Centredness Scale (Earthy 1998)	**Usability Maturity Model: Processes** (Earthy 1999)	**UUMM** (Lethbridge 2007)
Level X: Unrecognised	Level 0: Incomplete	
Level A: Recognised	Level 1: Performed	Level 1: Haphazard
Level B: Considered	Level 2: Managed	Level 2: Defined input from users and usability awareness
Level C: Implemented	Level 3: Established	Level 3: Iterative interactions with users and design for usability
Level D: Integrated	Level 4: Predictable	Level 4: Controlled and measured involvement of users
Level E: Institutionalized	Level 5: Optimising	Level 5: Continually improving usability

While studying the perceptions and practices of usability in OSS, Terry et al. (2010) highlight that HCI methods should be reframed on the basis of the social relationship between OSS developers and users. The authors stress the *"importance of the direct social relationship between developers and users in addressing usability issues."* OSS quality-related issues differ from those of closed proprietary software, especially in terms of quality control, quality assurance techniques, risk assessment, testing, and usability (Benson et al. 2004; Crowston et al. 2003 and Çetin and Göktürk 2007). None of the existing scales, however, describe the usability maturity of software in terms of issues that are specifically related to open source projects. Furthermore, the attributes used in these models have not been empirically validated. OSS is an important area of research where a lot of work needs to be performed in order to establish a comprehensive strategy for the evaluation of usability maturity. We thus need distinct usability criteria to assess OSS usability. Consequently, our work expands upon the existing concepts to complete the research gap in OSS usability assessment studies by proposing *A Usability Maturity Model for Open Source Software Projects*.

Chapter 3

3 Users’ Perspective of Open Source Usability: An Empirical Study

In recent years, the number of OSS users has increased, especially those who are non-technical users. Accordingly, many individuals believe that the OSS market share could increase tremendously provided OSS had systems that were easier to use. Although examples of usable open source software exist, most researchers acknowledge that OSS can be made more usable. This chapter presents an empirical investigation that studies the impact of several key usability factors from the perspective of end users. The resulting research model establishes the relationship between the usability factors from the perspective of users and OSS usability. Specifically, it utilizes a data set of 102 OSS users from 13 open source projects of various sizes. The results of this study provide empirical evidence indicating that these key factors play a significant role in improving OSS usability (Raza et al. 2010-a).

3.1 Usability Factors from the Perspective of Users: Literature Review of Concepts

“A user is a person who uses a computer or Internet service. A user may have a user account that identifies the user by a username” (Wikipedia-a)**.** In this work, we study and empirically analyze the impact of five key usability factors, including User Expectations, Usability Bug Reporting, Interactive Help Features, Usability Learning and Usability Guidelines.

User expectations *“refer to the consistency that users expect from products”* (Wikipedia-b)**.** Sampson (2007) questions whether or not OSS developers consult a usability expert or explore any of the standards in making their products more usable. He refers to the same dilemma of OSS developers who consider themselves as end users of their product, *“too often the team presumes that all the skills required are already present among team members.”* Twidale and Nichols (2005) explore usability related issues in open source software development, concluding that *“scarcity of expertise, bug reporting and* *classification, and heterogeneity in usability discussions"* are areas on which OSS developers need to focus. Hence, developers need to address users' expectations and requirements for their software to receive acceptance. However, Bodker et al. (2007) maintain that there is a gap between the OSS *"developer-users"* and their potential users. Specifically, they posit that enabling the employees of organizations and institutions to create the right requirements of OSS systems can be a way *"to avoid developer-centric systems that perform poorly in terms of real-world usability."* Nichols and Twidale (2005) also highlight the existence of a traditional approach by OSS developers in developing software for technically adept users. Nevertheless, they observe that with the increasing involvement of non-technical users in the OSS community, software developers are starting to realize the importance of usability and hence the requirement to ensure that new users find the software usable and adaptable. Therefore, OSS architects, designers and developers need to realize that they are not the ultimate users of their applications.

The second key factor, usability bug reporting, refers to design-time or runtime errors in software that are specific to and reported by the user. Zhao and Deek (2006) highlight the problem that users encounter when reporting OSS errors. These authors observe that when an average user wants to report an error, particularly one that is related to usability, s/he does not know how to do it effectively. Observing a similar problem, Nichols and Twidale (2006) identify one of the prominent usability challenges, stating that the *"difficulties that a user may experience with a Graphical User Interface may not be easy to describe textually."* Specifically, they maintain that there is a bias in treating usability bugs as compared to functionality bugs. Usability issues, as expected, are more subjective in nature and are more controversial; for example, a user interface (UI) element may be more confusing to some people than it is to others. Such issues could prolong the discussion of analyzing and fixing usability bugs, making these issues difficult to describe. Çetin et al. (2007) observe that due to *"the lack of a suitable usability reporting interface,"* many usability issues that occur are not reported. These authors propose that one of the ways to improve OSS usability is through effective feedback from end users, which can be achieved by providing users with an easy and convenient way to report the software errors that they encounter.

*"In computer science, interactive refers to software which accepts and responds to input from humans—for example, data or commands"(*Wikipedia-c). The addition of interactive help features in software that dynamically address user problems would contribute towards making software more usable. However, there are challenges involved in designing such features. First, in order to provide users with help, it is necessary to know exactly what type of help users are likely to request. Furthermore, such help features need to be designed in order to provide help to users with all levels of expertise, ranging from laymen to experts. This task is difficult to accomplish, as Shneiderman (2000) asserts that the process of designing software for an expert user is already difficult, so designing software for the average user is even more challenging. Although the reduced cost of hardware and computer accessories provides access for a greater number of people, the interface and information design is yet to appeal to all users. Viorres et al. (2007) argue that the majority of disabled users prefer to use proprietary software due to its greater accessibility and assistive technology, despite the fact that OSS claims to have the *"right to access for all."*

"Learning is acquiring new knowledge, behaviors, skills, values, preferences or understanding, and may involve synthesizing different types of information" (Wikipedia-d)**.** Formal usability learning by software architects, designers and developers can be an acknowledgement of the problem as well as a part of the solution. The practice of HCI, which is commonly referred to as Usability Engineering, is necessary for software developers to understand the perspective of users. Rusu et al. (2008) highlight the importance of HCI education for software professionals. They argue that in computer science (CS) design courses, HCI issues are generally given a secondary level importance. Additionally, Zhao and Deek (2006) suggest that OSS users also need to be trained so that they are able to perform usability inspections in an effective and efficient way.

A guideline is *"any document that aims to streamline particular processes according to a set routine"* (Wikipedia-e)**.** Nichols and Twidale (2006) maintain that Human Interface Guidelines (HIGs) represent instructions as to how OSS systems should be made more usable. Hedberg et al. (2007) propose the incorporation of usability guidelines, the active participation of usability experts in OSS projects, as well as the addition of usability testing and bug reporting. In order to understand the challenges related to usability and quality assurance in OSS, in-depth empirical research is necessary. According to Çetin and Göktürk (2008), there is no consensus on usability guidelines among OSS developers and usability experts. Iivari et al. (2008) believe that the growing user population of OSS is more interested in usable systems than they are in their development.

3.2 Research Model and Hypotheses

This work presents a research model for analyzing the relationship between key usability factors and open source software usability, as shown in Figure 3.1. The model derives its theoretical foundations by combining previous work in OSS usability, SE and HCI; it includes five key usability factors: *Users' Expectations, Usability Bug Reporting, Interactive Help Features, Usability Learning and Usability Guidelines*. The dependent variable of this study is *OSS Usability*, and the five independent variables are referred to as *"Usability Factors"* hereafter. Overall, the objective of this study is to investigate the answer to the following question: "*do key usability factors have an impact on OSS usability from the perspective of end users?"*

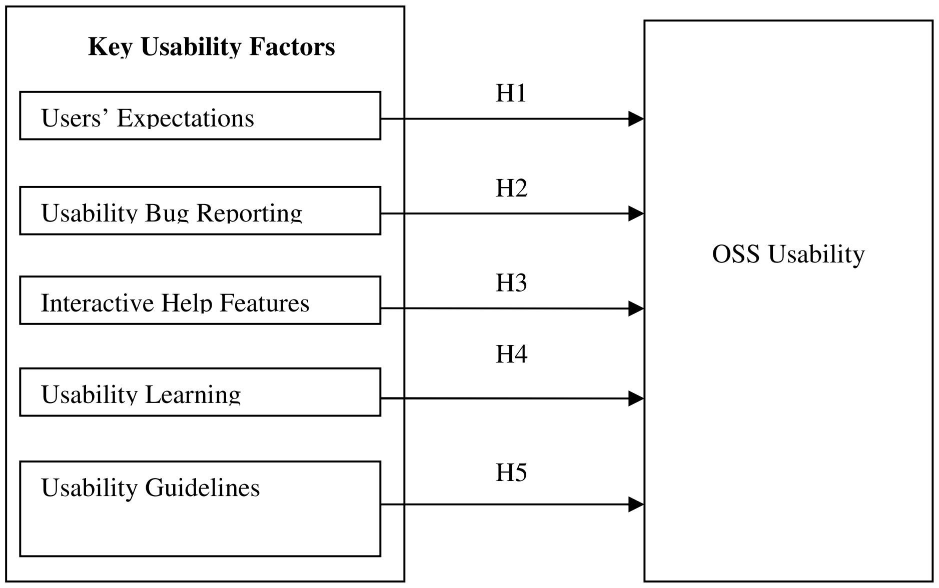


Figure 3.1: Research Model (Users' Perspective)

This work empirically investigates the association between these five usability factors and OSS usability. In this study, each relationship between the five independent variables and OSS usability is examined. The multiple linear regression equation of the model is as follows:

$$\text{OSS Usability} = f_0 + f_1 v_1 + f_2 v_2 + f_3 v_3 + f_4 v_4 + f_5 v_5 \quad (3.1)$$

where f_0, f_1, f_2, f_3, f_4 and f_5 are the coefficients and v_1, v_2, v_3, v_4 and v_5 are the five independent variables. In order to empirically investigate the research question, the five hypotheses are derived as presented in Table 3.1

Table 3.1: Research Model Hypotheses (Users' Perspective)

Hypothesis #	**Statement**
H1	The software developers' understanding of user expectations and requirements is positively related to improving usability in OSS.
H2	Convenient usability bug reporting has a positive impact on usability in OSS.
H3	Interactive help features in software have a positive impact on usability in OSS.
H4	The designers' knowledge of usability and user-centered design methods is positively related to improved software.
H5	Usability guidelines for the developers help to improve OSS usability.

3.3 Research Methodology

Open source software projects deal with different categories of applications, such as Database, Desktop Environment, Education, Finance, Games / Entertainment, and Networking. In order to collect the data, we sent emails from users' mailing lists to OSS users of 13 different projects on sourceforge.net. The projects differed in size and ranged from small-scale to large-scale. The questionnaires, which are contained in Table 3.2 to Table 3.7, were sent to the end users of projects with activity levels of 90% and above in the categories of Games / Entertainment, Database, Education, Office/Business and Scientific/Engineering as shown in Figure 3.2.

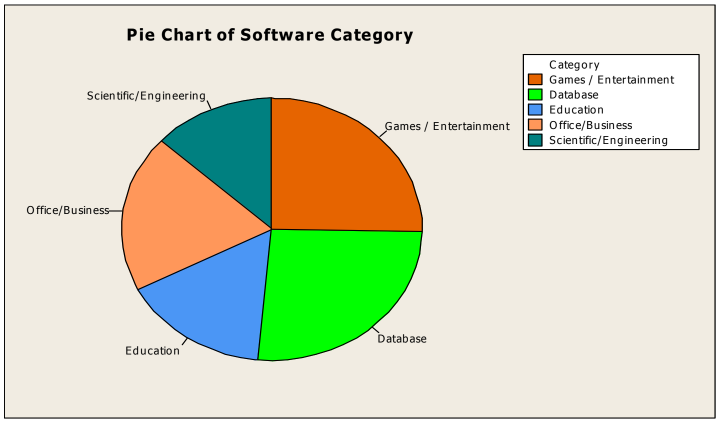


Figure 3.2: Respondents' Distribution

We assured the participants that our survey was confidential and that their identity would not be disclosed. However, in order to support our data analysis of user experience, we asked the respondents to reveal their experience with computers. Unlike the mandatory questions related to OSS usability, this particular question was optional. Out of the 102 responses that we received, 101 individuals chose to respond to this question. Among these responses, two respondents categorized themselves as novice computer users, ten considered themselves as average computer users and eighty-nine of the respondents believed that they were experienced users, as demonstrated in Figure 3.3. These statistics indicate the dominance of experienced computer users in the OSS arena.

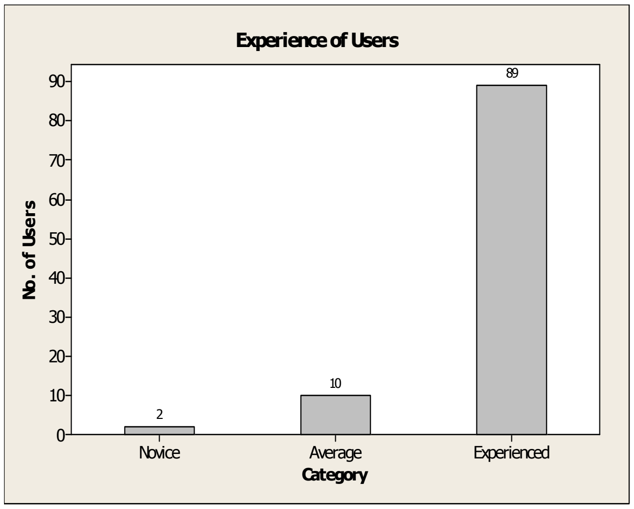


Figure 3.3: Experience of Users

3.3.1 Data Collection and the Measuring Instrument

In this study, we collected data on the key usability factors and the perceived level of usability by OSS users. The measuring instruments presented in Table 3.2 to Table 3.7 were used to learn the perceived level of OSS usability as well as the extent to which these usability factors were important for the users of the OSS projects. Specifically, we used twenty separate items to measure the independent variables and four items to measure the perspective of OSS users regarding usability. The questionnaire required respondents to indicate the extent of their agreement or disagreement with statements using a five-point Likert Scale. For all of the items associated with each variable, the scale ranged from "Strongly Agree" (1) to "Strongly Disagree" (5). The four items for each independent variable were designed to measure the extent to which the variable is practiced within each project. The items for all five usability factors were labeled sequentially in Table 3.2 to Table 3.6 and numbered one through twenty. Additionally, the dependent variable, OSS Usability, was also measured on the multi-item, five-point Likert Scale. The items were specifically designed to collect measures for this variable and are labeled sequentially from one through four in Table 3.7.

Table 3.2: Users' Expectations Measurement Instrument

1. The user interface of OSS should follow the standards and norms of proprietary software in order to make it easy to use.
2. I believe OSS is not meant for novice non-technical users.
3. Formal feedback from users is missing and thus needed in an OSS environment.
4. Proprietary software addresses users' expectations and requirements better than OSS.

Table 3.3: Usability Bug Reporting and Fixing Measurement Instrument

5. Ease of reporting errors in software would increase my level of satisfaction.
6. I never report an error, so usability bug reporting does not affect me.
7. User training is required for effective usability inspections.
8. Usability bugs reflect users' expectations; therefore, they need to be fixed based on priority.

Table 3.4: Interactive Help Features Measurement Instrument

9. I believe interactive help would increase the ease of use in open source software.
10. A novice user only requires the basic features of software, so interactive help features would not have much impact on his/her usage.
11. Users of OSS are technically sophisticated; they do not need interactive help.
12. Interactive help would increase the learnability of OSS.

Table 3.5: Usability Learning Measurement Instrument

13. OSS developers must learn how to incorporate user requirements and usability aspects in their software designs.
14. The lack of usability knowledge is the main cause of poor usability of OSS systems.
15. The realization that a software system is for end users rather than for the developers is more important than formal usability learning.
16. OSS developers should use quantifiable metrics to measure software usability objectively and effectively.

Table 3.6: Usability Guidelines for OSS Developers Measurement Instrument

17. There should be a standardized user interface and usability guidelines that OSS developers should follow in their designs.
18. Usability guidelines should act as a standard with which software should be inspected.
19. The strict implementation of usability guidelines will restrict the freedom of OSS developers.
20. Standardized usability guidelines are impracticable in an OSS environment.

Table 3.7: OSS Usability Measurement Instrument

1. OSS with standardized usability features will help users to compare the usability of different software.
2. Since poor usability is a major hurdle, the improved usability of OSS systems will result in users switching from proprietary software to OSS.
3. Usable software with satisfied users guarantees its success.
4. Open source software with improved usability and adaptability for less technical and novice users will benefit all users.

3.3.2 Reliability and Validity Analysis of the Measuring Instrument

The reliability and validity of a measurement are two integral features of an empirical study. Reliability indicates the reproducibility of a measurement, whereas validity refers to the agreement between the experimental value of a measurement and its true value.

The multiple-item measurement scales of the five usability factors are evaluated for reliability using an internal-consistency analysis, which was performed using the coefficient alpha (Cronbach 1951). In our analysis, the coefficient alpha ranges from 0.56 to 0.60, as shown in Table 3.8. van de Ven and Ferry (1980) state that a reliability coefficient of 0.55 or higher is satisfactory, and Osterhof (2001) suggests that 0.60 or higher is satisfactory. Therefore, based on our analysis, the variable items developed for this empirical investigation are reliable.

Table 3.8: Coefficient Alpha and Principal Component Analysis of variables (Users' Perspective)

Usability Factors	**Item no.**	**Coefficient α**	**PCA Eigen value**
Users' Expectations	1 – 4	0.60	2.34
Usability Bug Reporting	5 - 8	0.56	2.12
Interactive Help Features	9 - 12	0.57	2.07
Usability Learning	13 - 16	0.57	1.23
Usability Guidelines	17 - 20	0.60	1.02

According to Campbell and Fiske (1959), convergent validity occurs in a given assembly when the scale items are correlated and move in the same direction. The Principal Component Analysis (PCA) is performed for all five usability factors (Comrey and Lee 1992), as reported in Table 3.8. Kaiser (1970) maintains that the Eigen Value is used as a reference point to observe the construct validity with principal component analysis. We used the Eigen Value One-criterion, also known as the Kaiser Criterion (Kaiser 1960, Stevens 1986), which indicates that any component having an Eigen Value greater than one is retained. Eigen-value analysis reveals that all five variables completely form a single factor. Therefore, based on our statistical analysis, the convergent validity of our measuring instrument can be regarded as sufficient.

3.3.3 Data Analysis Procedure

Using different statistical techniques, we analyze the research model and the significance of hypotheses H1-H5. The analysis occurs in three phases; in Phase I, we use normal distribution tests and parametric statistics, whereas in Phase II, we use non-parametric statistics. Due to the relatively small sample size, both parametric and non-parametric statistical approaches are used to reduce the threat to external validity. Since our measuring instruments have multiple items for all five independent variables as well as for the dependent variable, as shown in Tables 3.2 to 3.7, the respondents' ratings are added together in order to obtain a composite value for each of them. Using parametric statistics, tests are conducted for hypotheses H1-H5 in order to determine the Pearson correlation coefficient. Alternatively, the Spearman correlation coefficient tests involve non-parametric statistics for hypotheses H1-H5. To deal with the limitations of a relatively small sample size and to increase the reliability of the results, hypotheses H1-H5 are tested using the Partial Least Square (PLS) technique in Phase III. According to Fornell and Bookstein (1982) as well as Joreskog and Wold (1982), the PLS technique is helpful in dealing with issues such as complexity, non-normal distribution, low theoretical information, and small sample size. All statistical calculations are performed using minitab-15[1].

3.4 Hypotheses Testing and Results

To test hypotheses H1-H5 of the research model, as shown in Figure 3.1, parametric statistics are used to examine the Pearson correlation coefficient between individual independent variables, the usability factors, and the dependent variable, OSS usability. The results of the statistical calculations for the Pearson correlation coefficient are displayed in Table 3.9. *"In statistical hypothesis testing, the p-value is the probability of obtaining a test statistic. The lower the p-value, the less likely the result is if the null*

[1] ***Minitab*** *is a statistics software package , and is often used in conjunction with the implementation of Six Sigma, CMMI and other statistics-based process improvement methods. Minitab is available in 7 different languages.*

hypothesis is true, and consequently the more "significant" the result is, in the sense of statistical significance" (Wikipedia-f).

The Pearson correlation coefficient between the users' expectations and OSS usability is positive (0.22) at $P < 0.05$, and hence, hypothesis H1 is justified. For H2, the relationship between usability bug reporting and OSS usability, the Pearson correlation coefficient is 0.37 at $P < 0.05$, and hence, it is found to be significant as well. Furthermore, the hypothesis H3 is accepted based on the Pearson correlation coefficient of 0.22 at $P < 0.05$, which represents the relationship between the interactive help features and OSS usability. The positive correlation coefficient of 0.49 at $P < 0.05$ is also observed between OSS usability and usability learning, which means that H4 is accepted. However, hypothesis H5, which denotes the relationship between usability guidelines and OSS usability, yields a Pearson correlation coefficient of 0.11 at $P = 0.27$, and thus, this hypothesis is statistically insignificant and consequently, it is rejected. Hence, as observed and reported, hypotheses H1, H2, H3 and H4 are found to be statistically significant and are accepted, whereas H5 is not supported and is therefore rejected.

In the second phase, non-parametric statistical testing is conducted by examining the Spearman correlation coefficient between the individual independent variables, the usability factors, and the dependent variable, OSS usability, as displayed in Table 3.9. First, the Spearman correlation coefficient between the users' expectations and OSS usability is found to be positive (0.28) at $P < 0.05$, and hence, hypothesis H1 is justified. For hypothesis H2, which examines the relationship between usability bug reporting and OSS usability, the Spearman correlation coefficient of 0.37 is observed at $P < 0.05$, and hence, this hypothesis is significant. Moreover, the hypothesis H3 is accepted based on the Spearman correlation coefficient of 0.27 at $P < 0.05$, demonstrating a statistically significant relationship between interactive help features and OSS usability. A positive Spearman correlation coefficient of 0.42 at $P < 0.05$ is observed for the fourth hypothesis, which represents the relationship between OSS usability and usability learning, indicating that H4 is also accepted. For hypothesis H5, which involves usability guidelines and OSS usability, the Spearman correlation coefficient of 0.03 is observed at $P =0.79$. Since no significant relationship is found between the usability guidelines and OSS usability, H5 is rejected.

Hence, as observed and reported, H1, H2, H3 and H4 are found to be statistically significant and are accepted, whereas H5 is not supported and hence rejected in both parametric and non parametric analysis.

Table 3.9: Hypotheses testing using parametric and non-parametric correlation coefficients (Users' Perspective)

Hypothesis	**Usability Factor**	**Pearson Correlation Coefficient**	**Spearman Correlation Coefficient**
H1	Users' Expectations	0.22*	0.28*
H2	Usability Bug Reporting	0.37*	0.37*
H3	Interactive Help Features	0.22*	0.27*
H4	Usability Learning	0.49*	0.42*
H5	Usability Guidelines	0.11**	0.03**

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$.

In the third phase, the PLS technique is used to perform the cross validation of results obtained in Phase I and Phase II. Specifically, this method examines the direction and significance of hypotheses H1–H5. In the PLS technique, the dependent variable of the research model, OSS usability, is the response variable, and the independent key usability factors are the predicators. The test results containing the observed values of the path coefficients, R^2 and the F-ratio are shown in Table 3.10. The first variable, users' expectations, is significant at $P < 0.05$, with a path coefficient of 0.22, an R^2 value of 0.49 and an F-ratio of 5.15. Furthermore, the variable of usability bug reporting has a path coefficient of 0.58, an R^2 value of 0.35 and an F-ratio of 15.57, and hence, it is significant at $P < 0.05$. The next variable, interactive help features, has the same direction as those proposed in hypothesis H3, with a path coefficient of 0.46, an R^2 value of 0.50 and an F-ratio of 5.26 at $P < 0.05$, and so it is also significant. Moreover, the variable of usability learning conforms to hypothesis H4, with a path coefficient of 0.62, an R^2value of 0.47 and an F-ratio of 32.82 at $P < 0.05$. Finally, the last variable, usability guidelines, has a path coefficient of 0.22, an R^2 value of 0.12 and an F-ratio of 1.20 at $P = 0.27$. Hence, in this phase, as in Phases I and II, hypothesis H5, which deals with usability guidelines and OSS usability, is not found to be statistically significant at $P > 0.05$, and thus, it is rejected.

Table 3.10: Hypotheses testing using Partial Least Square regression (Users' Perspective)

Hypothesis	**Usability Factor**	**Path Coefficient**	**R^2**	**F- Ratio**
H1	Users' Expectations	0.22	0.49	5.15*
H2	Usability Bug Reporting	0.58	0.35	15.57*
H3	Interactive Help Features	0.46	0.50	5.26*
H4	Usability Learning	0.62	0.47	32.82*
H5	Usability Guidelines	0.22	0.12	1.20**

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The multiple linear regression equation of our research model is depicted in Equation 3.1. For this statistical test, the testing process includes regression analysis, which yields the values of the model coefficients and their direction of association. In this case, OSS usability is considered as the response variable and the usability factors are the predicators. As shown in Table 3.11, the path coefficients for all five variables are positive, whereas the t-statistics for four out of five variables, including users' expectations, usability bug reporting, interactive help features and usability learning, are statistically significant at $P < 0.05$. In contrast, the t-value for the usability guidelines is observed as 0.40 at $P = 0.68$, thus making the variable of usability guidelines statistically insignificant in this research model.

Table 3.11: Multiple Linear Regression Analysis of the User Model

Model coefficient Name	**Model coefficient**	**Coefficient value**	**t-value**
Users' Expectations	f_1	0.22	2.55*
Usability Bug Reporting	f_2	0.20	1.79*
Interactive Help Features	f_3	0.19	1.69*
Usability Learning	f_4	0.38	4.40*
Usability Guidelines	f_5	0.04	0.40**
Constant	f_0	3.92	0.45*

Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The R^2 value and the adjusted R^2 value for the overall research model are observed as 0.32 and 0.29 respectively, with an F-ratio of 9.10, which is significant at $P < 0.05$.

Recapping Equation 3.1 by inserting the model coefficient values, we get:

$$\text{OSS Usability} = 3.92 + 0.22v_1 + 0.20v_2 + 0.19v_3 + 0.38v_4 + 0.04v_5$$

where v_1, v_2, v_3, v_4 and v_5 are the five independent variables.

3.5 Critical Analysis of the Study

In recent years, the increasing use of open source software results from factors such as easy, and for the most part, free access to the internet. However, many individuals believe that closed proprietary software is still superior in areas such as post-release maintenance, quality control and management. According to Aberdour (2007), "*sustainable communities, code modularity, project management, and test process management*" are the major factors upon which the quality of OSS relies. Twidale (2005) stresses the need for "*participatory usability involving the coordination of end users and developers*." With the popularity of OSS among organizations as well as among common novice users, the OSS community is no longer limited to *"technically adept"* individuals. Hence, the requirements and expectations of OSS are not the same as they were a decade ago, when software developers were considered to be the only OSS users. Çetin et al. (2007) identify users, customers and developers as the major groups involved in OSS bug reporting. Specifically, these authors emphasize the need for using experts' opinions in order to improve the usability of open source software. In conclusion, they state that in order to improve OSS usability, factors such as "*usability reports, usability laboratories, usability experts and companies specializing in usability*" need to be given serious consideration. Iivari et al. (2008) studied the usability of OSS in a business context, maintaining that "*there probably will be a need for HCI methods and tools, which are particularly tailored to fit the OSS development context, as well as for HCI specialists contributing to the development*."

In their empirical study for measuring the success of OSS projects, Lee et al. (2009) argue that "*OSS use was significantly influenced by software quality and user satisfaction*." Furthermore, software developers need to communicate more effectively with the target audience of their product in order to achieve a greater understanding of the users' perspective (Koppelman 2006). Accordingly, this study maintains that in order to achieve user satisfaction, software designers and developers must understand the expectations and requirements of end users. In fact, this study's empirical investigation demonstrated a positive relationship between users' expectations and OSS usability. Therefore, software developers should consider the expectations of users as a key issue necessary for improving the usability of their projects.

However, there are several challenges associated with increasing the usability of open source software. First, software usability is a subjective concept, and thus it cannot be directly measured. Additionally, many users experience difficulty in reporting usability errors. Nichols and Twidale (2006) argue that it is challenging for a user to describe and hence report the difficulties s/he faces in the Graphical User Interface (GUI) of software. Similarly, seventy-eight percent of the respondents in our survey agree that a greater degree of convenience in reporting software errors would increase their level of satisfaction. Thus, software designers and developers need to increase the ease of reporting usability related errors. Additionally, our empirical investigation reveals a positive association between usability bug reporting and OSS usability. Thus, usability bug reporting and fixing is considered to be a key issue for improving the usability of OSS projects.

Among the variety of software users, elderly individuals, children and people with disabilities have unique expectations and challenges. For disabled users, closed proprietary software is preferable to OSS, mainly because of its greater accessibility and level of assistance (Viorres et al. 2007). Thus, in order to increase the level of acceptance for OSS among different categories of users, the help features of software need to be made interactive and dynamic. In fact, seventy-one percent of the respondents in our survey agree that interactive help would increase their learnability of open source software.

Benson et al. (2004) observe that quite often, OSS is "*created by engineers for engineers*." Hence, in order to increase the market share and accessibility of OSS, developers need to enhance the appropriateness of usability tactics, thus making their systems more usable. Similarly, our empirical investigation reveals that usability learning has a positive impact on OSS usability. For instance, seventy-nine percent of our survey's respondents agree that OSS developers must learn how to incorporate users' requirements and usability aspects in their software designs. In addition, our empirical analyses confirm the existence of a positive relationship between usability learning and OSS usability. Therefore, usability learning is one of the key issues required for improving the usability of OSS projects.

Nichols and Twidale (2006) discuss human interface guidelines that not only prevent confusion about usability issues but also may be considered as an authority on courses of action that should be taken. However, sixty-three percent of the respondents in our survey believe that the strict implementation of usability guidelines limits the freedom of OSS developers. Similarly, the results of our empirical study, including both parametric and non-parametric statistical analyses as well as PLS and multiple regression testing, do not support a positive relationship between usability guidelines and OSS usability. Therefore, our study does not show a positive association between usability guidelines and OSS usability.

3.5.1 Limitations & Threats to External Validity

As in the case of any empirical investigation, this study has certain limitations. Easterbrooks et al. (2007) refer to construct validity, internal validity, external validity and reliability as four criteria of validity in an empirical study. In most cases, the researcher's ability to generalize the experimental outcome to industrial practice is generally limited by threats to external validity (Wohlin et al. 2000), which is the case with this study. We took specific measures to support external validity, including our use of a random sampling technique that selects respondents from the general population. Additionally, we retrieved data from the most active and well-known OSS reporting website, sourceforge.net, which includes a large number of projects.

The increased popularity of empirical methodology in software engineering has also raised concerns of an ethical nature (Faden et al. 1986, Katz 1972). However, our study adhered to the recommended ethical principles in order to ensure that the empirical investigation would not violate any of the recommended experimental ethics.

Furthermore, another aspect of validity concerns whether or not the study results correspond to previous findings. Our work involved the selection of five independent variables that related to the dependent variable of OSS usability. While there are other key factors that influence usability, the scope of this study was restricted to the area of open source software from the perspective of end users. Our other studies, which consider the perspectives of developers, contributors and the industry, incorporate more contributing factors, including experts' opinion, incremental design approaches, usability testing, knowledge of user-centered design methods, user feedback and usability design techniques. Consequently, our work considers many more usability factors.

Another limitation of this study involves its relatively small sample size. Although we sent our survey to a large number of OSS users who subscribed to thirteen different categories of software, we only received 102 responses. Consequently, the relatively small number of responses was a potential threat to the external validity. However, we followed the appropriate research procedures by conducting and reporting tests in order to improve the reliability and validity of the study, and certain measures were also taken to increase the external validity.

This study investigates the effects of key factors on OSS usability from the end users' perspective. The results of the empirical analysis show that the factors in our research model assist in understanding and analyzing the end users' perception of OSS usability. Specifically, the results support the hypotheses that users' expectations, usability bug reporting and fixing, interactive help features and usability learning have a positive impact on the usability of an OSS project. However, we could not find any significant statistical support for the positive effect of usability guidelines on OSS usability. Nevertheless, our study can assist OSS designers and developers in enhancing their understanding of the relationships between the key factors and the usability of their projects. In addition, these factors should be considered by the OSS development community for addressing usability issues in their projects. Lastly, this empirical investigation provides some justification for considering these key factors as a measuring instrument for our maturity model, which assesses the usability of open source software projects.

Chapter 4

4 Open Source Software Usability: An Empirical Evaluation from the Developers' Perspective

In application software, the satisfaction of a target user makes the software more acceptable. Although OSS does not necessarily have poor usability, as there are strong examples of usable open source software, the usability of OSS can be significantly improved. Accordingly, this chapter presents an empirical investigation that studies the way in which several key factors impact OSS usability from the developers' point of view. Focusing on this perspective, the research model in this empirical investigation examines the relationship between usability factors and OSS usability. A data set of 106 OSS developers from 18 open source projects of various sizes is used to study the research model (Raza et al. 2010-b)

4.1 Usability Factors from the Perspective of Developers: A Literature Survey

"A software developer is a person or organization concerned with facets of the software development process. This person may contribute to the overview of the project on the application level rather than component level or individual programming tasks" (Wikipedia-g). Additionally, *"perspective in theory of cognition is the choice of a context or a reference (or the result of this choice) from which to sense, categorize, measure or codify experience, cohesively forming a coherent belief, typically for comparing with another"* (Wikipedia-h). Markov (2003) argues that usability is about *"total user experience,"* rather than only about the user interface, as researchers commonly but incorrectly assume. Although not every OSS has a poor user interface, the usability of OSS projects generally require improvement. Nichols and Twidale (2005) maintain that OSS is growing and has developed a reputation of being reliable, efficient and functional. However, novice computer users still prefer to use proprietary software, especially because it has better usability than OSS. These authors discuss usability in applications such as word processors and web mail servers, which are essentially aimed at serving a

novice user. In addition, they maintain that because OSS has fewer resources than closed proprietary software, an OSS project would take longer to mature in comparison to those of closed proprietary software. In the OSS culture, the process of coding starts earlier and the design refinement depends on constant reviews. Hence, in order to improve OSS usability, the interface design should be completed before the start of the coding process, thus maintaining consistency. On the other hand, Viorres et al. (2007) refer to various reasons why software developers prefer OSS to closed proprietary software. These reasons include educational factors, re-usability and the development of a strong reputation. However, these authors also highlight concerns about software usability in OSS, including complex installation and maintenance of development tools, non-adherence to backwards compatibility and limited documentation. Hedberg et al. (2007) propose the adaptation of proven methods in the OSS environment in order to ensure a higher quality of software and to address usability issues. For instance, Holzinger et al. (2006) discuss a user-centered system developed at the clinical department of Dermatology at the Medical University Hospital in Graz. This system not only improved the existing system, but it also helped elderly people to overcome their reluctance with using computers.

In this work, we empirically study the impact of several factors on OSS usability, including users' requirements, usability experts' opinion, an incremental design approach, usability testing and the knowledge of user-centered design methods.

A user requirement refers to *"a statement that identifies a necessary attribute, capability, characteristic, or quality of a system in order for it to have value and utility to a user*" (Wikipedia-i). The ability of OSS developers to understand user requirements and expectations is an important issue that needs to be given serious consideration. Pemberton (2004) recognizes the fact that software developers take a different intuitive approach than that of end users; he observes that while developers are creating software, they are normally satisfied with its usability and interface, especially since they intend to develop the software for themselves, and they do not consider users' requirements or expectations. Similarly, Nichols and Twidale (2006) also argue that developers generally do not understand the needs and expectations of end users, which may lead to poor usability in OSS. Çetin and Göktürk (2008) maintain that the main objective of OSS is software development through collaboration and cooperation. Traditionally, OSS users have had technical and computer-oriented backgrounds, and so they have required less effort to use OSS systems such as Apache, GNU C Compilers and Bash Shell. However, as OSS has become more popular, there is a greater need to have usable systems. Koppelman and Van Dijk (2006) stress that software developers should not simply rely on their own experiences and instincts. Rather, they should learn how to communicate with users in order to better understand their requirements and expectations.

"An expert, more generally, is a person with extensive knowledge or ability based on research, experience, or occupation and in a particular area of study" (Wikipedia– j). In the area of software usability, the importance of experts' opinions cannot be undermined. Large commercial organizations generally employ such experts to address usability issues in their projects. However, their representation is generally missing in OSS projects, probably due to the voluntary work environment of OSS. Nichols and Twidale (2005) explain why usability experts are not generally involved in OSS projects; firstly, it is because there are fewer such experts in the OSS world. Secondly, these experts are not *"incentivised by the OSS approach in the way that many hackers are,"* and finally, they are not *"welcomed into OSS projects."* Hedberg et al. (2007) emphasize the need for usability experts to contribute in OSS projects, and they show concern regarding experts' lack of participation in OSS development.

Incremental Design Approach involves the introduction of advanced software features to users in an incremental way that would make them more comfortable with these features. *"The basic idea is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental), allowing the developer to take advantage of what was learned during the development of earlier portions or versions of the system"* (Wikipedia-k). For example, gaming software uses this approach all of the time, allowing the users to manage advanced levels in an incremental and gradual fashion. OSS developers need to realize that their target audiences may include novice users, for whom the software application would be more adaptable if advanced features were introduced in a gradual and progressive way. For example, Yunwen and Kishida (2003) highlight the need for modularized software system designs to enable the end users to encounter difficult levels gradually and progressively. In particular, they believe that modularized OSS system architecture designs with the progressive introduction of difficult and advanced features would attract more users. Similarly, Aberdour (2007) suggests that code modularity is a convenient way to add new features in software. Specifically, it reduces the complexity of codes and allows different programmers to extend the program by working in parallel but not interfering with other programmers.

The fourth factor, usability testing, is necessary for improving the usability of software. According to Kaner (2006), *"software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test."* Çetin and Göktürk (2008) highlight the importance of testing and measurement by stating that *"one can't improve what is not measured."* Nichols and Twidale (2005) maintain that usability problems are neither easy to specify nor convenient to be fixed, particularly considering the virtual boundaries of OSS, where most developers do their work autonomously. In their another study, Nichols and Twidale (2006) observe that usability issues, as expected, are more subjective in nature and more debatable; for instance, a user interface element may be more confusing to some people than it is to others. These issues could increase the difficulty of analyzing and fixing usability bugs. In order to increase user participation in analyzing and fixing usability errors, it is necessary to make usability reporting easier and to eradicate the *"cultural, technical and usability barriers."* Unlike functionality errors, where duplication in bug reporting does not help, a high instance of usability bug reporting can help in prioritizing the usability related errors. Viorres et al. (2007) highlight a few potential areas of improvement for OSS usability, such as enhancing bug reporting facilities in software, improving the analysis of usability errors through the application of HCI principles, and supporting argumentation for resolving such issues.

"User-centered design (UCD) is a design philosophy and a process in which the needs, wants, and limitations of end users of a product are given extensive attention at each stage of the design process" (Wikipedia-l). In order to increase their understanding of users' perspectives, students of the Software Engineering and Computer Science

disciplines should be taught how to address user-centric issues in their software development projects. Moreover, they should be trained to realize that finding a solution to a particular programming problem is not the ultimate goal; rather, they should create designs that meet the expectations of end users. Faulkner and Culwin (2000) observe that HCI and SE educators have always held different beliefs. Although the importance of HCI has been increasingly realized, especially in literature and education, these authors suggest that software engineering should adopt HCI as the underlying principle for systems development. Moreover, they believe that the aim of usability engineering education must be to ensure that effectiveness, efficiency and user satisfaction are present in software. As a result, an HCI specialist needs to assist software developers through the unification of knowledge and vocabulary for both professionals. However, Rosson et al. (2004) maintain that the main challenge in teaching usability engineering is to provide realistic projects for students, so that meaningful issues could be addressed in a manageable period of time.

4.2 Hypotheses Model

In this study, we present a research model to analyze the relationship between five unique key usability factors and the usability of open source software; the model, which is based on an empirical investigation of the association between these factors and usability, is depicted in Figure 4.1. Although there could be other key factors that influence OSS usability, we chose our variables from extensive literature survey, to keep the scope of this study within open source software and the OSS developers' points of view. In this investigation, our aim is to answer the following research question:

How can OSS developers improve software usability?

There are five independent variables and one dependent variable in this research model. The five independent variables include users' requirements, usability experts' opinion, incremental design approach, usability testing and knowledge of user-centered design methods. As in the last study, the dependent variable is OSS usability.

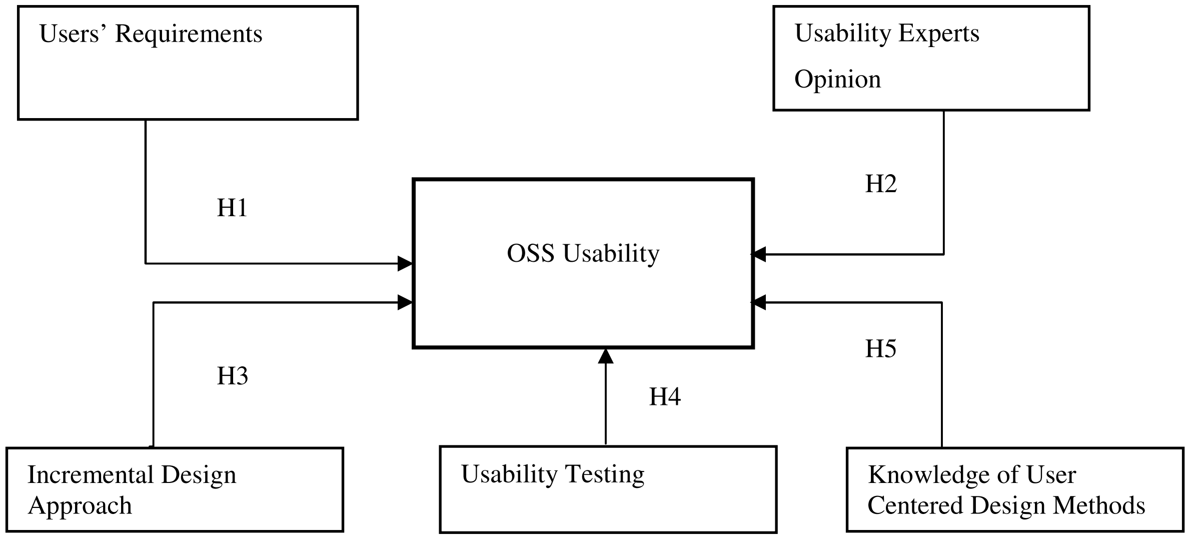

Users' Requirements
Usability Experts Opinion
H1
H2
OSS Usability
H3
H4
H5
Incremental Design Approach
Usability Testing
Knowledge of User Centered Design Methods

Figure 4.1: Research Model (Developers' Perspective)

In discerning the association between usability factors and OSS usability, this work examines the way in which the five independent variables affect OSS usability. The multiple linear regression equation of the model is as follows:

$$\text{OSS Usability} = \alpha_0 + \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3 + \alpha_4 v_4 + \alpha_5 v_5 \qquad (4.1)$$

where $\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_4$ and α_5 are the coefficients, and v_1, v_2, v_3, v_4 and v_5 are the five independent variables. In order to empirically investigate the research question, five hypotheses are presented in Table 4.1

Table 4.1: Research Model Hypotheses (Developers' Perspective)

Hypothesis #	**Statement**
H1	The software developers' understanding of the users' requirements is positively related to improving usability in OSS.
H2	The software developer's ability to seek usability experts' opinion is positively related to improving usability in OSS.
H3	An incremental approach in OSS design plays a positive role in improving usability in OSS.
H4	Usability testing has a positive impact on usability in OSS.
H5	The knowledge of User-Centered Design (UCD) methods is positively related to improving usability in software.

4.3 Research Methodology

Open source software projects deal with different categories of applications, including Communications, Database, Desktop Environment, Education, Finance, Games / Entertainment, and Networking. Accordingly, we sent personalized emails to OSS developers of different projects on sourceforge.net. The projects differed in size and ranged from small-scale to large-scale projects. For our study, we selected the projects that had an activity level of 90% or greater. Subsequently, we sent our questionnaires to OSS developers working on projects in the categories of Database, Desktop Environment, Development, Testing, Communications, Games / Entertainment, Education, Finance and Enterprise, as shown in Figure 4.2.

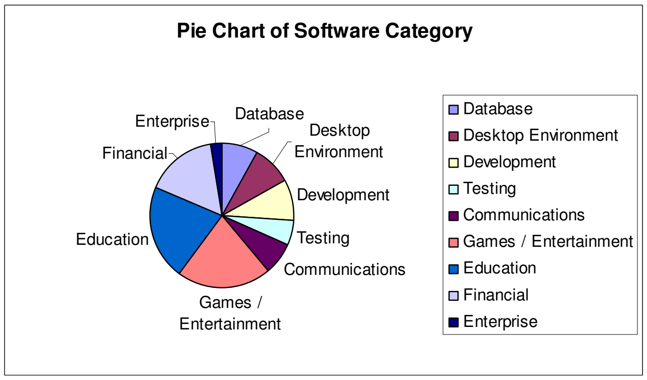


Figure 4.2 Pie chart of Software Category

We assured the participants that our survey did not require their identity and would not be recorded. However, in order to support our data analysis of the developers and of the project size, we asked them to discuss their OSS development experience and their development team size. Nevertheless, these two questions were optional, unlike the mandatory questions related to OSS usability. In total, we received 106 responses, and 104 of the participants responded to these two questions. As seen in Figure 4.3, 63 respondents had less than five years of OSS development experience, 31 participants had five to ten years of experience and ten respondents had more than 10 years of experience.

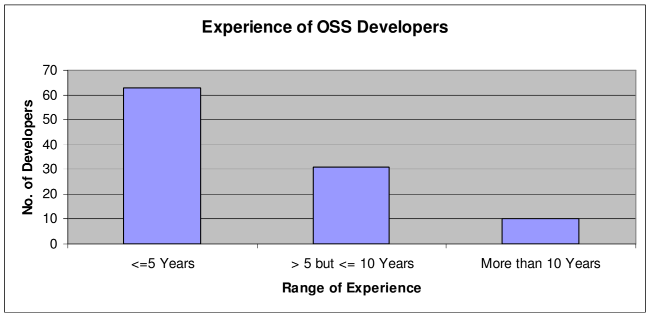


Figure 4.3: Experience of OSS Developers

Furthermore, 56 respondents had less than 10 developers in their project, 27 participants had between 10 and 20 team members as developers and 21 had more than 20 members in their development team. These statistics are represented in Figure 4.4.

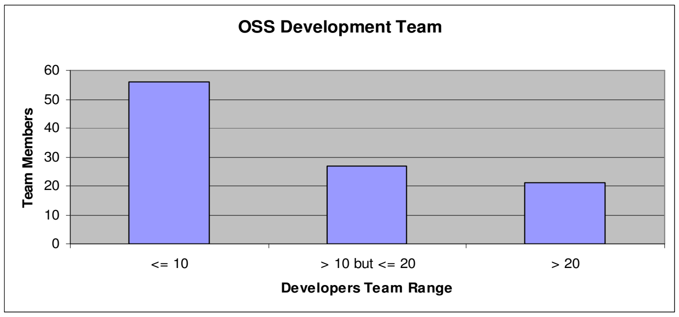


Figure 4.4: OSS Development Teams

The above statistics are presented to reflect the experience of the respondents as well as the size of the OSS project they belong to.

4.3.1 Data Collection and the Measuring Instrument

The questionnaires presented in Tables 4.2 to 4.7 require respondents to indicate the extent of their agreement or disagreement with statements using a five-point Likert Scale. The Likert Scale ranges from “Strongly Agree” (1) to “Strongly Dis-agree” (5) for all items associated with each variable. Four statements are presented for each of the six variables; these statements describe the specific key factor and its related issues, and they are designed to collect measures on the extent to which the variable is practiced within each project. Thus, we have used twenty separate items to measure the independent variables and four items to measure the dependent variable. Although not much work in this area has been conducted, we have reviewed previous studies on the subject of OSS usability in order to construct a comprehensive list of measuring factors.

Table 4.2: Users’ Requirements Measurement Instrument

1. Users’ requirements help in increasing software usability.
2. The code contributors’ understanding of community expectations supports software usability.
3. Encouraging community feedback before and after the formal release of every major software version is vital in improving software usability.
4. Recording users’ profiles is crucial in understanding their requirements and expectations and hence supports OSS usability.

Table 4.3: Usability Experts' Opinions Measurement Instrument

5. Usability features can be incorporated more effectively if experts' opinions are encouraged during every life-cycle phase.
6. Seeking usability experts' opinions will compromise the freedom of OSS developers.
7. OSS designs that are based on usability experts' opinions have standard usability norms but lack innovation.
8. Usability experts' opinions are equally important and applicable for OSS as they are for closed proprietary software.

Table 4.4: Incremental Design Approach Measurement Instrument

9. Incremental increases in the difficulty level of software makes users feel more comfortable.
10. A novice user requires only the basic features of software.
11. The gradual introduction of advanced features will enhance the adaptability of software; however, this process is not always feasible.
12. Every user should gradually explore the advanced features of software.

Table 4.5: Usability Testing Measurement Instrument

13. Formal usability testing should be an integral part of the Software Testing Process.
14. Although software success is dependent on users' responses, usability-related bugs mostly reflect personal demands.
15. I will fix usability-related bugs only if I am convinced that the reported bug is worth fixing.
16. Usability bugs reflect users' requirements and expectations; therefore, they need to be fixed in terms of priority.

Table 4.6: Knowledge of UCD Methods Measurement Instrument

17. Computer Science and Software Engineering students that will be future software developers must learn how to incorporate usability aspects in their software designs.
18. The designing of a user-friendly GUI is an art that not every programmer can learn.
19. The CS/SE curriculum needs to be revised to reinforce the importance of user-centeredness in software designs.
20. Poor usability in OSS systems is not due to a lack of knowledge of user-centered design methods; it is because such designs are not implemented and the systems are not designed with people in mind.

Table 4.7: OSS Usability Measurement Instrument

1. Improving OSS usability will result in reducing the overall cost, the reporting of bugs and the software defects.
2. One reason for poor OSS usability is because OSS designers do not receive compensation; OSS designers should have some incentive, such as awards or recognition to motivate their development efforts.
3. A successful software project is one with usable software and satisfied users.
4. Software with improved usability and adaptability for novice users will benefit all users.

4.3.2 Reliability and Validity Analysis of the Measuring Instrument

The two most important aspects of precision in survey studies are reliability and validity. In empirical studies, the most commonly used approaches involve conducting reliability and validity analyses of the measuring instruments. The reliability of the multiple-item measurement scales for the five usability factors is evaluated by using internal-consistency analysis, which is performed using the coefficient alpha (Cronbach 1951). In our analysis, the coefficient alpha ranges from 0.55 to 0.67, as shown in Table 4.8. van de Ven and Ferry (1980) state that a reliability coefficient of 0.55 or higher is satisfactory, and Osterhof (2001) suggests that 0.60 or higher is satisfactory. Therefore, based on the standards in the literature, the variable items developed for this empirical study are reliable.

Table 4.8: Coefficient Alpha and Principal Component Analysis of variables (Developers' Perspective)

Usability Factors	**Item no.**	**Coefficient α**	**PCA Eigen value**
Users' Requirements	1 – 4	0.67	2.19
Usability Experts' Opinion	5 – 8	0.64	1.22
Incremental Design Approach	9 – 12	0.55	1.08
Usability Testing	13 – 16	0.55	0.99
Knowledge of UCD Methods	17 – 20	0.59	1.05

Convergent validity, according to Campbell and Fiske (1959), occurs when the scale items in a given construct move in the same direction and, therefore, correlate strongly with one another. The principal component analysis (Comrey and Lee 1992), which provides a measure of convergent analysis, is performed for all five key usability factors, as reported in Table 4.8. We have used the Eigen Value (Kaiser 1970) as a reference point for observing the construct validity using principal component analysis. Specifically, we have used the Eigen Value One Criterion, also known as the Kaiser Criterion (Kaiser 1960, Stevens 1986), which means that any component having an Eigen Value greater than one is to be retained. In our study, Eigen Value analysis reveals that four out of five variables completely form a single factor, whereas the Eigen Value for the usability testing is 0.99, which is very close to the threshold of 1.0. Therefore, the convergent validity of the variables is sufficient.

4.3.3 Data Analysis Procedure

We have analyzed the research model and the significance of hypotheses H1 to H5 through different statistical techniques. This analysis occurs in three phases; in Phase I, we have used normal distribution tests and parametric statistics, whereas in Phase II, we have implemented non-parametric statistics. Due to a relatively small sample size, both parametric as well as non-parametric statistical approaches are used to reduce the threats to external validity. Since our measuring instrument has multiple items for all five independent variables as well as for the dependent variable, we have added the ratings from the respondents to obtain a composite value for each variable. For testing hypotheses H1 to H5, parametric statistics are used to determine the Pearson correlation coefficient and non-parametric statistics are used for determining the Spearman correlation coefficient. In Phase III, the hypotheses H1 to H5 are tested using Partial Least Square (PLS) technique, which minimizes the limitations of a relatively small sample size and increases the reliability of the results. According to Fornell and Bookstein (1982) as well as Joreskog and Wold (1982), the PLS technique supports issues such as complexity, non-normal distribution, low theoretical information, and small sample size. All of these statistical calculations are performed using Minitab-15.

4.4 Empirical Analysis and Findings

To test the hypotheses of the research model, parametric statistics are used to calculate the Pearson correlation coefficient between the individual independent variables, the key usability factors, and the dependent variable, OSS usability. The results of the statistical calculations for the Pearson correlation coefficients are displayed in Table 4.9.

The Pearson correlation coefficient between the users' requirements and OSS usability is positive, with a value of 0.26 at $P < 0.05$, and hence, it justifies hypothesis H1. For the second hypothesis, the Pearson correlation coefficient between experts' opinion and OSS usability is 0.08 at $P = 0.39$, and thus, it is statistically insignificant at $P < 0.05$ and therefore rejected. The hypothesis H3 is accepted based on the Pearson correlation coefficient of 0.27 at $P < 0.05$, which measures the relationship between the incremental design approach and OSS usability. Furthermore, the positive correlation coefficient of 0.34 at $P < 0.05$ occurs between OSS usability and usability testing, which indicates that H4 is accepted. Finally, H5 is found to be statistically significant and thus accepted after obtaining the Pearson correlation coefficient of 0.44 at $P < 0.05$ between the knowledge of UCD methods and OSS usability. Hence, the hypotheses H1, H3, H4, and H5 are statistically significant and are accepted, whereas H2 is not supported and is therefore rejected.

Alternatively, non-parametric statistical testing is conducted by examining the Spearman correlation coefficient between the individual independent variables and the dependent variable. As is the case with the Pearson correlation coefficient, the results of the statistical calculations for the Spearman correlation coefficient are displayed in Table 4.9. The Spearman correlation coefficient between the users' requirements and OSS usability is positive, with a value of 0.48 at $P < 0.05$, and hence, the hypothesis H1 is justified. For hypothesis H2, the Spearman correlation coefficient of 0.12 is observed at $P = 0.21$; hence, at $P < 0.05$, no significant relationship is found between experts' opinions and OSS usability. However, the hypothesis H3 is accepted based on the Spearman correlation coefficient of 0.42 at $P < 0.05$, which indicates the relationship between the incremental design approach and OSS usability. Moreover, a positive Spearman correlation coefficient of 0.39 at $P < 0.05$ occurs between OSS usability and usability testing, thus indicating that H4 is accepted. Lastly, hypothesis H5 is statistically significant and thus accepted based on a Spearman correlation coefficient of 0.48 at $P < 0.05$, which occurs between the knowledge of UCD methods and OSS usability.

Therefore, H1, H3, H4, and H5 are found statistically significant and are accepted, whereas H2 is not supported and is hence rejected in both the parametric and the non-parametric analyses.

Table 4.9: Hypotheses testing using parametric and non-parametric correlation coefficients (Developers' Perspective)

Hypothesis	**Usability Factor**	**Pearson Correlation Coefficient**	**Spearman Correlation Coefficient**
H1	Users' Requirements	0.26*	0.48*
H2	Usability Experts' Opinion	0.08**	0.12**
H3	Incremental Design Approach	0.27*	0.42*
H4	Usability Testing	0.34*	0.39*
H5	Knowledge of UCD Methods	0.44*	0.48*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$.

The PLS technique is used to perform the cross validation of results obtained in Phases I and II. Specifically, this test examines the direction and significance of hypotheses H1 to H5. In the PLS method, the dependent variable of our research model, OSS usability, is used as the response variable, and the independent variables, the usability factors, function as predicators. The test results containing the observed values of the path coefficient, R^2, and the F-ratio, are shown in Table 4.10. The variable of Users' Requirements is significant at $P < 0.05$, with a path coefficient of 0.30, an R^2 value of 0.07 and an F-ratio of 7.78. The next variable, usability experts' opinion, has a path coefficient of 0.13 with an R^2 value of 0.01 and an F-ratio of 0.74; accordingly, this hypothesis is insignificant at $P < 0.05$ with an observed value of $P = 0.39$. Incremental design approach has the same direction as proposed in hypothesis H3 with path coefficient of 0.24, an R^2 value of 0.07 and an F-ratio of 8.43 at $P < 0.05$. Furthermore, usability testing conforms to the hypothesis H4, with the following values: path coefficient= 0.31, R^2= 0.11 and F-ratio= 13.41 at $P < 0.05$. The last variable, the knowledge of UCD methods, has a path coefficient of 0.45, an R^2 value of 0.19 and an F-ratio of 24.89 at $P < 0.05$, and thus, it is agrees with H5. Hence, in this phase, as in Phase I and Phase II, the hypothesis H2, which indicates the relationship between usability experts' opinion and OSS usability, is not statistically significant at $P < 0.05$.

Table 4.10: Hypotheses testing using PLS regression (Developers' Perspective)

Hypothesis	**Usability Factor**	**Path Coefficient**	**R^2**	**F- Ratio**
H1	Users' Requirements	0.30	0.07	7.78*
H2	Usability Experts' Opinion	0.13	0.01	0.74**
H3	Incremental Design approach	0.24	0.07	8.43*
H4	Usability Testing	0.31	0.11	13.41*
H5	Knowledge of UCD Methods	0.45	0.19	24.89*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

Multiple Linear Regression equation of the research model, depicted in Equation 4.1, tests our research model, thereby providing empirical evidence that the studied key factors play a significant role in improving open source software usability. Accordingly, the testing process consists of conducting regression analysis and obtaining the values of the model coefficients and their direction of association. In this case, OSS usability is considered as the response variable and the key factors are the predicators, as shown in Table 4.11. The path coefficient of four out of the five independent variables, users' requirements, incremental design approach, usability testing and knowledge of user centered design methods, are positive, and their t-statistics are also considered statistically significant at $P < 0.05$. The path coefficient of usability experts' opinion is negative, and accordingly, the negative t-statistics at $P > 0.05$ make this variable statistically insignificant.

Table 4.11: Multiple Linear Regression Analysis of the Research Model (Developers' Perspective)

Model coefficient Name	**Model coefficient**	**Coefficient value**	**t-value**
Users' Requirements	α_1	0.28	2.80*
Usability Experts' Opinion	α_2	-0.01	-0.04**
Incremental Design Approach	α_3	0.12	1.22*
Usability Testing	α_4	0.11	1.10*
Knowledge of UCD Methods	α_5	0.35	3.74*
Constant	α_0	1.80	1.00*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The R^2 and adjusted R^2 value of the overall research model are observed as 0.29 and 0.26 respectively with an F-ratio of 8.33, which is significant at $P < 0.05$.

Recapping Equation 4.1 by inserting the model coefficient values, we get:

$$\text{OSS Usability} = 1.80 + 0.28v_1 - 0.01v_2 + 0.12v_3 + 0.11v_4 + 0.35v_5$$

where v_1, v_2, v_3, v_4 and v_5 are the five independent variables.

4.5 Discussion of the Results

The use of open source software has increased in recent years, mainly due to the availability of internet. Although it has been a common belief that OSS is utilized mainly by technically adept users, thus obscuring the boundaries between developers and users, there has been a significant increase in the number of novice and non-technical users. Due to the growing prevalence of novice users, issues relating to usability need to be given research priority. Accordingly, our study uses empirical investigation to enable OSS developers and project managers to understand the relationship between the key factors of our research model and the OSS usability process. The results provide empirical evidence and support for the hypotheses that several key factors play an important role in the institutionalization of usability within an OSS project.

In order for users to become satisfied with OSS, it is necessary to understand their expectations and requirements. OSS is no longer a "*reserved arena*" for technically adept users; novice and non-technical users from all over the world use open source software. Koppelman and Van Dijk (2006) explain that the knowledge of end user requirements and expectations can be achieved through increased communication between software developers and their target users, rather than software developers using their instincts and opinions for creating usable software. Similarly, 87 percent of our respondents agree that an understanding of user requirements helps in improving OSS usability. Additionally, empirical investigation found a positive relationship between users' requirements and OSS usability. Therefore, we consider the variable of users' requirements, a key issue for OSS developers in improving the usability of their projects.

In software development, the role of usability experts cannot be understated, especially in application software, where end users are the direct audiences. In proprietary software development, large organizations hire experts to share their opinion for making software more usable and acceptable to end users. However, because work in OSS is voluntary and there are fewer resources in OSS development, there are not many usability experts active in the OSS field. In fact, Nichols and Twidale (2005) explain that these experts might not find themselves "*welcomed into OSS projects.*" Nevertheless, our statistical findings do not significantly support the positive association between usability experts' opinions and OSS usability. In the parametric and non parametric statistical analysis, as well as in PLS and multiple regression testing, the results were not deemed statistically significant, as demonstrated in Tables 4.9 to 4.11. Therefore, our study has not been able to show a positive association between Usability Experts' Opinion and OSS usability.

The gradual and incremental introduction of advanced features in software assists in making users more comfortable. Specifically, it increases the acceptability and adaptability of the application. Yunwen and Kishida (2003) advocate the use of modularized system design, where users encounter each difficulty level gradually and progressively. For instance, gaming software uses an incremental approach in their designs, where users need to complete one level in order to progress to the more difficult levels. The use of this approach in all designs can make software more accommodating for a common novice user. In fact, 69 percent of the respondents in our survey agree that gradual introduction of advanced features in software would enhance its adaptability. Additionally, our research study demonstrated that the incremental design approach has a positive impact on OSS usability. Therefore, we consider the Incremental Design Approach as a key attribute for improving OSS usability.

Software testing is an integral part of the software life cycle. Holzinger (2005) advocates the necessity of usability testing early in the software life cycle, maintaining that *"the earlier critical design flaws are detected, the more likely they can be corrected."* However, there are two challenges to usability testing. First, since software usability is a subjective matter, it cannot be directly measured, and secondly, users encounter difficulties in reporting errors. 72 percent of the respondents in our survey agree that formal usability testing should be an integral part of the software testing procedure. Similarly, the findings of our empirical investigation also confirm a positive association between usability testing and OSS usability. Thus, despite its challenges, Usability Testing is a key issue for improving the usability of OSS projects.

As future software managers and developers, students of Computer Science and Software Engineering need to enhance their realization of the extent to which usable systems are crucial. In particular, they should be taught that it is more important for systems to meet users' expectations than it is to find a programming solution to a problem. It is essential for students to incorporate usability features early in their designs in order to facilitate the maintenance of their projects. In our empirical investigation, we found a positive relationship between the knowledge of user-centered design methods and OSS usability. Based on our study, the education of UCD methods could be a part of long-term solution for improving software usability; it would be equally beneficial to both OSS and closed proprietary software organizations. Therefore, we consider the Knowledge of UCD methods as one of the key factors for improving the usability of OSS projects.

4.5.1 Limitations of the Study & Threats to External Validity

There are several empirical methods for investigating both software engineering processes and products, including surveys, experiments, metrics, case studies, and field studies (Singer and Vinson 2002). All of these empirical investigations are subject to certain limitations, which is the case with this study.

Threats to external validity are conditions that limit the researcher's ability to generalize the results of his/her experiments to industrial practice (Wohlin et al. 2000). In this study, specific measures have been taken to support external validity; for example, a random sampling technique has been used to select the respondents from the population. Additionally, we retrieved the data from the most active and well-known OSS reporting website, sourceforge.net, which contains a significant number of projects.

The increased popularity of empirical methodology in software engineering has also raised concerns with ethical issues (Faden et al. 1986, Katz 1972). However, in this study, we have followed the recommended ethical principles to ensure that the empirical investigation would not violate any form of the recommended experimental ethics.

Another aspect of validity is concerned with whether or not the study reports results that correspond to previous findings. We have made several efforts to maintain validity in this case; first of all, we selected five independent variables to relate to the dependent variable of OSS usability. Other contributing factors, such as usability bug reporting, interactive help features, usability learning, usability guidelines, user feedback, and usability design techniques are considered in our other studies, which measure the effect these factors on usability from the perspective of users, contributors and the industry. Consequently, we consider several more usability factors in our empirical investigations.

Another limitation of this study is its relatively small sample size. Although we sent our survey to a significant number of OSS developers in 18 different projects of software, we received only 106 responses. The relatively small number of respondents poses a potential threat to the external validity of this study. Although the proposed approach has some potential to threaten external validity, we have followed the appropriate research procedures by conducting and reporting tests to improve the reliability and validity of the study.

In this study, we examined the effect of key factors on OSS usability and found answer to the research question stated in this empirical investigation. The study results demonstrated that the key factors in our research model assist in improving OSS usability. Specifically, the hypotheses that OSS usability is positively affected by Users' Requirements, Incremental Design Approach, Usability Testing and Knowledge of UCD Methods are demonstrated in the investigation. However, we could not find any significant statistical support that Experts' Opinion is positively related to OSS usability. Overall, this study will enable OSS development teams to improve their understanding of the relationships between the stated key factors and the usability of their projects.

This empirical investigation provides us with justification for considering several key factors as a measuring instrument for our usability maturity model. Additionally, this study is one of the series of four studies that we have conducted in parallel, regarding OSS usability from the perspective of users, contributors and the industry.

Chapter 5

5 Contributors' Viewpoint of Open Source Software Usability: An Empirical Study

The number of users of open source software (OSS) is practically unlimited, and the software quality is determined by the experience of the end users. Because of these two factors, usability is an even more critical quality attribute for OSS than it is for proprietary software. Accordingly, the research model of this chapter establishes the relationship between OSS usability and several key usability factors from the perspective of contributors. In order to study the model, a data set of 78 OSS contributors, including architects, designers, developers, testers and users from 22 open source projects of varied size has been acquired. The results of this study provide empirical evidence indicating that these specified key factors play a significant role in improving OSS usability (Raza and Capretz 2010-a).

5.1 Usability Factors: Literature Review of Concepts

In the context of OSS, contributors are individuals who contribute towards project development. Contribution roles may differ, as they can include architects, designers, developers, testers or users.

According to Wikipedia, *"feedback describes the situation when output from (or information about the result of) an event or phenomenon in the past will influence an occurrence or occurrences of the same (i.e. same defined) event / phenomenon (or the continuation / development of the original phenomenon) in the present or future"* (Wikipedia-m). Benson et al. (2004) observe that in an OSS environment, the "*feedback cycle with real users*" is absent. As a result, these authors list several challenges to improving OSS quality, including the communication gap between the OSS developers and the users, the lack of target user profiles and the developers' degree of responsibility. Similarly, Bouktif et al. (2006) identify the "*lack of automated feedback*" about the system's quality as one of the major weaknesses in the open source environment. They also propose "*feedback-driven communication service*" to send feedback to developers after each commitment. In her empirical study about user participation in OSS projects, Iivari (2009-b) acknowledges "*informative, consultative and participative roles for users.*" Bevan (2006) also considers feedback from users as the most popular way of improving software usability; however he believes "*it leaves open the risk of inadequate final usability.*" Consequently, Crowston et al. (2003) recommend the collection of user feedback by building a survey into the software that measures users' satisfaction levels.

"The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships between them" (Wikipedia-n). The process of addressing usability issues at the software architecture level, which is the foundation of the entire software, can save a significant amount of subsequent improvement effort. However, to perform this endeavor, it is necessary to understand user requirements and to accurately set priorities while understanding that usability is an important quality attribute, which is true for both closed software and OSS. Golden et al. (2005) identify a common practice of software architects when they "*assume that usability issues that arise during user testing can be handled with localized modifications.*" Specifically, they observe that this practice entails high costs, especially when usability-related problems occur at the time of testing, as it may require redesigning, and in the worst case scenario, even reconstructing the entire system. Viorres et al. (2007) believe that the involvement of end users during the design and development of OSS eliminate most of the challenges in the software lifecycle. They advocate the need for applying HCI principles in the design processes of OSS to make use of their full potential. Not unlike the other authors, Bevan (2008) supports the utilization of human-centered design resources in the earlier stages of the software life cycle.

"Any system that is designed for people should be easy to use, easy to learn, and useful for the users" (Wikipedia-o). In their survey, Zhao and Elbaum (2000) observe that the quality techniques used in OSS differ from traditional software practices. They conclude that, unlike systematic activities in traditional software development, OSS quality assurance is dependent on "*revisions, enhancements and corrections*" by users who are

actively involved in the projects. Hedberg et al. (2007) believe that although multiple meanings have been attached to the user-centered design methodology, all of these meanings "*emphasize the importance of understanding the user, his/her tasks or work practices and the context of use*." Elaborating on the work of others, Folmer and Bosch (2004) list three types of user feedback: usability testing that requires user feedback on a typical system task, usability evaluation questionnaires, and usability surveys, especially those that gather the opinion of usability experts or developers regarding the extent to which a user interface complies with standard usability norms.

"Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test" (Kaner 2006). Çetin and Göktürk (2008) maintain that OSS developers need to recognize the usability level of their projects. Although traditionally, software testing consumes considerable time, there is a limited amount of formal testing conducted by OSS developers. Aberdour (2007) contrasts the "*formal and structured testing*" that is typical in closed software development with the "*unstructured and informal testing*" common in OSS development. Similarly, Hedberg et al. (2007) emphasize that formal software testing requires more attention in OSS projects through sufficient planning, thereby ensuring that errors are caught before the software is released.

In describing user documentation, Wikipedia states that *"typically, the user documentation describes each feature of the program, and assists the user in realizing these features. A good user document can also go so far as to provide thorough troubleshooting assistance. It is very important for user documents to not be confusing and for them to be up to date"* (Wikipedia-p). Overall, the lack of OSS documentation, which follows the formal life cycle of software development, may be due, in part, to the fact that developers are solely focused on software development and have such a thorough understanding of their code that they do not need formal documentation. Nevertheless, formal documentation assists new users who wish to understand and adapt to a particular system. Aberdour (2007) highlights the lack of documentation for OSS projects, especially in comparison to the extensive documentation in closed proprietary software. Accordingly, Aberdour emphasizes the necessity of complete documentation in

OSS projects. Among other indicators of OSS quality and success, Crowston et al. (2003) refer to the quality of codes and documentation, user ratings, downloads, and the reuse of code.

5.2 Research Organization and Hypotheses

In this study, we present a study model for analyzing the relationship between five key usability factors and the usability of open source software. Although there could be other key factors that influence OSS usability, we chose our variables to keep the scope of this study within open source software and the OSS contributors' points of view. The theoretical model that will undergo empirical testing is shown in Figure 5.1.

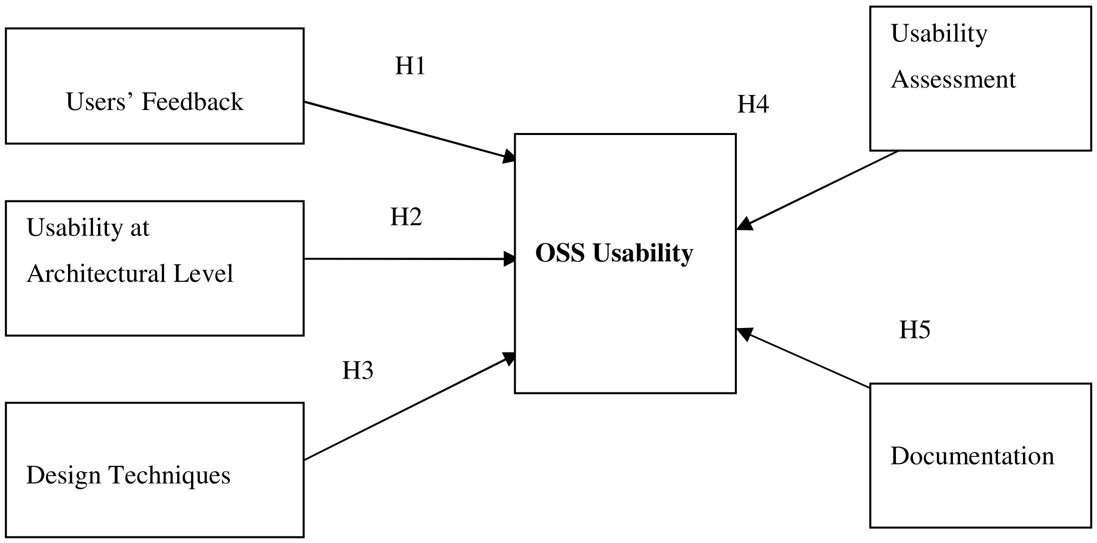


Figure 5.1: Study Model (Contributors' Viewpoint)

In particular, we examine the effect of five independent variables on OSS usability, which is the dependent variable in this model. Our aim is to investigate the answer to the following research question:

How can OSS usability be improved from the contributors' perspective?

The five independent variables include Users' Feedback, Usability at the Architectural Level, Design Techniques, Usability Assessment and Documentation, while the dependent variable is OSS usability. The multiple linear regression equation of the model is as follows:

$$\text{OSS Usability} = \beta_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 + \beta_4 v_4 + \beta_5 v_5 \qquad (5.1)$$

where β_0, β_1, β_2, β_3, β_4 and β_5 are the coefficients and v_1, v_2, v_3, v_4 and v_5 are the five independent variables. In order to empirically investigate the research question, five hypotheses are presented in Table 5.1

Table 5.1: Study Model Hypotheses (Contributors' Viewpoint)

Hypothesis #	**Statement**
H1	Formal feedback from users has a positive impact on OSS usability.
H2	The software designers' ability to address usability issues at the architectural level is positively related to improving usability in OSS.
H3	The designers' use of user-centered design techniques is positively related to improving software.
H4	The usability assessment and testing of software has a positive impact on OSS usability.
H5	Formal software documentation plays a positive role in improving OSS usability.

5.3 Research Methodology

Open source software projects deal with different categories of applications, including Database, Desktop Environment, Education, Finance, Games / Entertainment, and Networking. For our research, we sent personalized emails to OSS contributors of different projects, which differed in size and ranged from small-scale to large-scale. We sent our questionnaire to the contributors of projects in the categories of Communications, Database, Desktop Environment, Education, Formats and Protocols, Software Development, Finance and Games / Entertainment (115), as shown in Figure 5.2.

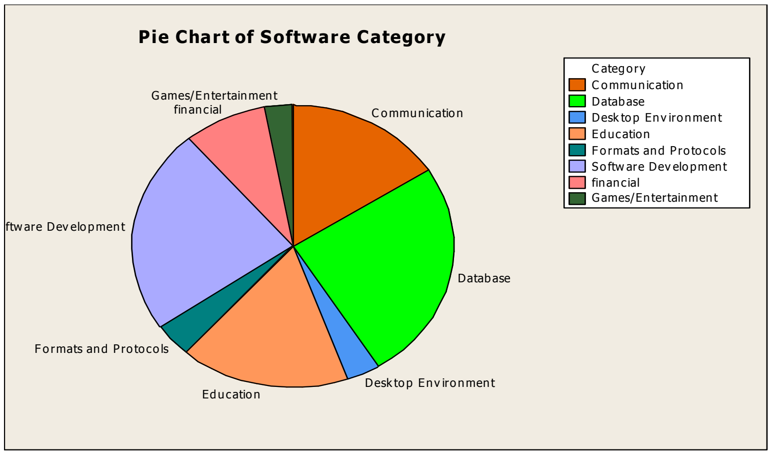


Figure 5.2: Participants' Distribution

We assured the participants that our survey did not require their identity and would not be recorded. However, the participants were required to disclose their specific project role, such as architect/designer, developer, tester or user, in order to support our data analysis of contributors and their participation in OSS projects. Out of 78 total respondents, 11 were architects/designers, 32 were developers, 2 were testers and 33 were users.

5.3.1 Data Collection and the Measuring Instrument

The questionnaires, presented in Tables 5.2 to 5.7, measure the perceived level of OSS usability as well as the extent to which these usability factors are important for the contributors of OSS projects. In particular, we have used twenty separate items to measure the independent variables and four items to measure the contributors' perspective of OSS usability. In order to obtain a comprehensive list of measuring factors, we reviewed previous studies on the subject of OSS usability. Specifically, we used a five-point Likert Scale to measure the extent to which each of these usability factors have been practiced in OSS projects or agreed upon by OSS contributors. The Likert scale ranges from "Strongly Agree" (1) to "Strongly Disagree" (5) for all items associated with each variable. Four items for each independent variable have been designed to collect measures on the extent to which the variable is practiced within each project; the items for all five usability factors are labeled sequentially in Tables 5.2 to 5.6 and are numbered as 1 through 20. Additionally, we measured the dependent variable, OSS Usability, on the multi-item, five-point Likert Scale. The items have been specifically designed for collecting measures for this variable and are labeled sequentially from 1 to 4 in Table 5.7.

Table 5.2: User's Feedback Measurement Instrument

1. User feedback is useful during every phase, including requirements, design, development, pre-release and post-release.
2. The quantification of user feedback and its contribution towards an OSS project is essential.
3. User feedback is more effective if it is recorded with a user's profile.
4. User feedback is required for application software only.

Table 5.3: Usability at Architectural Level Measurement Instrument

5. Incorporating usability at the architectural level saves subsequent modification efforts in terms of time and money.
6. It is impractical to incorporate usability at the architectural level.
7. Usability issues arise at the time of graphical user interface (GUI) development.
8. Usability at the architectural level is only an issue in large projects.

Table 5.4: Design Techniques Measurement Instrument

9. Standardized design techniques are needed for software usability.
10. Because of the various functionalities and contexts of use in software, standardized design techniques for usability are not feasible.
11. Standardized user interface guidelines are necessary for OSS developers.
12. User-centered design techniques can act as a set of criteria with which software may be inspected.

Table 5.5: Usability Assessment Measurement Instrument

13. The usability assessment of software is relevant in its context of use only.
14. Usability testing is important because of the distributed environment and the different cultural backgrounds of OSS users.
15. There is no need for formal usability assessment, as usability is already evaluated through developers' peer reviews and community feedback.
16. The usability assessment of an OSS project enhances its quality assurance, especially its reliability and acceptability.

Table 5.6: Documentation Measurement Instrument

17. The proper documentation of OSS projects increases the understandability and learnability of the software.
18. The documentation of OSS projects is impractical due to the voluntary work by OSS developers and the frequent release of versions.
19. With proper documentation, OSS becomes a better alternative to proprietary software.
20. Proper documentation increases the acceptability of OSS to general users and to organizations.

Table 5.7: OSS Usability Measurement Instrument

1. Since the quality of OSS relies heavily on user participation and peer reviews, usability issues need to be addressed more seriously.
2. OSS usability cannot be improved unless it is properly assessed and measured.
3. The success of software depends equally on its correct functionality and usability.
4. Usable software is more popular with elderly users, children and individuals with disabilities.

5.3.2 Reliability and Validity Analysis of the Measuring Instrument

In this empirical study, the reliability and validity of the measuring instruments are conducted by using the most common approaches. Specifically, the reliability of the multiple-item measurement scales for the five usability factors is evaluated by using internal-consistency analysis, which is performed using the coefficient alpha (Cronbach 1951). In our analysis, the coefficient alpha ranges from 0.64 to 0.74, as shown in Table 5.8. van de Ven and Ferry (1980) state that a reliability coefficient of 0.55 or higher is satisfactory, and Osterhof (2001) suggests that a value of 0.60 or higher is satisfactory. Therefore, the variable items developed for this empirical investigation are reliable.

Table 5.8: Coefficient Alpha and the PCA of variables (Contributors' Viewpoint)

Usability Factors	**Item No.**	**Coefficient α**	**PCA Eigen Value**
Users' Feedback	1 - 4	0.65	1.48
Usability at Architectural Level	5 - 8	0.74	1.21
Design Techniques	9 - 12	0.64	1.48
Usability Assessment	13 - 16	0.65	1.36
Documentation	17 - 20	0.68	1.51

The principal component analysis (Comrey and Lee 1992) for all five key usability factors in Table 5.8 provides a measure of convergent validity. According to Campbell and Fiske (1959), convergent validity occurs when the scale items are correlated and move in the same direction. We have used the Eigen Value (Kaiser 1970) as a reference point for observing the construct validity using principal component analysis. In this study, we used the Eigen value-one-criterion, also known as the Kaiser Criterion (Kaiser 1960, Stevens 1986), which indicates that any component having an Eigen Value greater than one is retained. Consequently, our Eigen Value analysis revealed that all five variables completely formed a single factor. Therefore, the convergent validity in our empirical study can be regarded as sufficient.

5.3.3 Data Analysis Procedure

We have used various statistical analysis techniques for analyzing the model and verifying the significance of hypotheses H1-H5. Initially, we divided the data analysis activity into three phases. In order to reduce the threats to external validity resulting from our small sample size, we used both parametric and non-parametric statistical methods. Before performing statistical analysis, we made some modifications to the data. Since the measuring instrument for all five of the independent variables and the dependent variable has multiple items, we added their ratings to obtain a composite score.

In Phase I, we conducted tests for hypotheses H1 to H5 using parametric statistics, such as the Pearson correlation coefficient, and in Phase II, we utilized non-parametric statistics, such as the Spearman correlation coefficient. Finally, Phase III entailed testing the hypotheses using the Partial Least Square (PLS) technique. The PLS technique is especially useful in situations involving complexity, non-normal distribution, low theoretical information, and small sample size (Fornell and Bookstein 1982, Joreskog and Wold 1982). Since small sample size is one of the major limitations to this study, we used the PLS technique to increase the reliability of the results. All statistical calculations were performed using Minitab–15 Software.

5.4 Study Results

For the first phase of this study, parametric statistics were used by examining the Pearson correlation coefficient between the individual independent variables, the key usability factors, and the dependent variable, OSS usability, in order to test hypotheses H1 to H5. The results of the statistical calculations for the Pearson correlation coefficient are reported in Table 5.9. The coefficient between user feedback and OSS usability is positive, with a value of 0.48 at $P < 0.05$, and hence, hypothesis H1 is justified. The Pearson correlation coefficient of 0.21 is observed at $P = 0.06$ for the relationship between usability at the architectural level and OSS usability, and hence, it is found insignificant at $P < 0.05$. Therefore, hypothesis H2, which involves usability at the architectural level and OSS usability, is rejected. Hypothesis H3 is accepted based on a Pearson correlation coefficient of 0.48 at $P < 0.05$, which involves the relationship between usability design techniques and OSS usability. The positive correlation coefficient of 0.36 at $P < 0.05$ occurs between OSS usability and usability assessment, which indicates that H4 is accepted. Finally, hypothesis H5 is found significant and accepted due to its Pearson correlation coefficient of 0.51 at $P < 0.05$, which occurs between documentation and OSS usability. Hence, the hypotheses H1, H3, H4, and H5 are found statistically significant and are accepted whereas H2 is not supported and is therefore rejected.

In Phase II, non-parametric statistical techniques are used by examining the Spearman correlation coefficient between the individual independent variables, the key usability factors, and the dependent variable, OSS usability, as shown in Table 5.9. The Spearman correlation coefficient between users' feedback and OSS usability is positive, with a value of 0.43 at $P < 0.05$, and hence, hypothesis H1 is justified. For hypothesis H2, the Spearman correlation coefficient of 0.29 occurs at $P = 0.01$; hence, at $P < 0.05$, a significant relationship is found between usability at the architectural level and OSS usability. Furthermore, hypothesis H3 is accepted based on a Spearman correlation coefficient of 0.48 at $P < 0.05$, which occurs between design techniques and OSS usability. The positive Spearman correlation coefficient of 0.32 at $P < 0.05$ is also observed between OSS usability and usability assessment, which indicates that H4 is accepted. Hypothesis H5 is also found significant and thus accepted after obtaining a Spearman correlation coefficient of 0.57 at $P < 0.05$, which exists between documentation and OSS usability improvement. Hence, all of the hypotheses, H1, H2, H3, H4 and H5, are found statistically significant and are accepted in the non-parametric analysis.

Table 5.9: Hypotheses testing using Pearson correlation and Spearman correlation coefficients (Contributors' Viewpoint)

Hypothesis	**Usability Factor**	**Pearson Correlation Coefficient**	**Spearman Correlation Coefficient**
H1	Users' Feedback	0.48*	0.43*
H2	Usability at the Architectural Level	0.21**	0.29*
H3	Design Techniques	0.48*	0.48*
H4	Usability Assessment	0.36*	0.32*
H5	Documentation	0.51*	0.57*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$.

In Phase III, the PLS technique is used to overcome some of the study's limitations and to perform cross-validation with the results observed using the approaches in Phases I and II. Specifically, the direction and significance of hypotheses H1 to H5 are examined. In the PLS method, the dependent variable of our study model, OSS usability, is considered as the response variable, and the independent variable, the key usability factors, are considered as predicators. The test results, which contain the observed values of the path coefficient, R^2 and the F-ratio, are demonstrated in Table 5.10. The users' feedback is observed to be significant at $P < 0.05$, with a path coefficient of 0.76, an R^2 value of 0.23 and an F-ratio of 22.68. Usability at the architectural level has path coefficient of 0.38, an R^2 value of 0.04 and an F-ratio of 3.59, which is found insignificant at $P < 0.05$, with an observed P value of 0.06. Usability design techniques have the same direction as proposed in the hypothesis H3, with a path coefficient of 1.03, an R^2 value of 0.23 and an F-ratio of 22.86 at $P < 0.05$. Similarly, the variable of usability assessment conforms to the hypothesis H4, with a path coefficient of 0.52, an R^2 value of 0.13 and an F-ratio of 11.39 at $P < 0.05$. Finally, the variable of documentation has a path coefficient of 1.08, an R^2 value of 0.26 and an F-ratio of 26.44 at $P < 0.05$, and, as a result, it also accords to H5. Hence in this phase, as is the case in Phase I, the hypothesis H2, which deals with usability at the architectural level and OSS usability improvement, is not statistically significant at $P < 0.05$.

Table 5.10: Hypotheses testing using PLS regression (Contributors' Viewpoint)

Hypothesis	**Usability Factor**	**Path Coefficient**	**R^2**	**F- Ratio**
H1	Users' Feedback	0.76	0.23	22.68*
H2	Usability at the Architectural Level	0.38	0.04	3.59**
H3	Design Techniques	1.03	0.23	22.86*
H4	Usability Assessment	0.52	0.13	11.39*
H5	Documentation	1.08	0.26	26.44*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

Our objective in study model testing is to provide empirical evidence that our key factors play a considerable role in improving open source software usability. In particular, the testing process consists of conducting regression analysis and reporting the values of the model coefficients and their direction of association. The multiple linear regression equation for our study model is depicted in Equation 5.1, where OSS usability is the response variable and the key factors are predicators. The regression analysis results of the study model are displayed in Table 5.11. Specifically, the path coefficient for four out of five variables, users' feedback, design techniques, usability assessment and documentation, are positive and their t-statistics are statistically significant at $P < 0.05$. However, the path coefficient for usability at the architectural level is negative, its negative t-statistics and its P value of 0.09 at $P > 0.05$ render this independent variable statistically insignificant.

Table 5.11: Multiple Linear Regression Analysis of the Study Model (Contributors' Viewpoint)

Model coefficient	**Model coefficient**	**Coefficient value**	**t-value**
Users' Feedback	β_1	0.32	3.13*
Usability at the Architectural Level	β_2	-0.16	-1.69**
Design Techniques	β_3	0.17	1.51*
Usability Assessment	β_4	0.19	2.16*
Documentation	β_5	0.47	5.02*
Constant	β_0	0.59	0.10*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The R^2 and the adjusted R^2 vales of the overall study model are 0.50 and 0.47 with an F-ratio of 14.24, which is significant at $P < 0.05$.

Recapping Equation 5.1 by inserting model coefficient values, we get:

$$\text{OSS Usability} = 0.59 + 0.32v_1 - 0.16v_2 + 0.17v_3 + 0.19v_4 + 0.47v_5$$

where v_1, v_2, v_3, v_4 and v_5 are the five independent variables.

5.5 Study Discussion

From the usability perspective, OSS is expected to encounter an increasingly challenging environment, with a diversity of users, both in terms of technical experience and cultural background, who possess unique needs, expectations and demands. Benson et al. (2004) emphasize the need for HCI professionals to enhance the acceptance and usability of OSS. According to Çetin and Göktürk (2008), OSS projects can only become usable through measurement and analysis.

Hedberg et al. (2007) state that it is essential to understand the user and the context of use as well as the active involvement of target users through their feedback at an earlier stage of the design process. Crowston et al. (2003) maintain that OSS relies heavily on its users' feedback for improving its quality. Accordingly, 82% of the respondents in our survey agree that user feedback is useful in every phase of the software lifecycle, including requirements, design, development, pre-release and post-release. Our empirical investigation also supports the hypothesis that user feedback has a positive impact on usability in OSS.

Golden et al. (2005) maintain that usability issues are not addressed at the software architecture design level, and, as a result, they identify the consequences of this omission, such as redesigning or re-architecting the whole system. Nakagawa et al. (2008) highlight the paucity of literature exploring how software architecture can influence OSS quality. Through a case study, they have shown that software architecture is positively associated with OSS quality. However, our parametric statistical analysis, PLS regression and multiple regression analyses do not support the positive relationship between the incorporation of usability issues at the architectural level and OSS usability.

Çetin and Gokturk (2007) identify the lack of user-centered design and usability problems in an OSS development environment. Specifically, they believe that there is a need for increased collaboration between designers and developers as well as an early contribution of usability experts in order to ensure the overall quality of a project. However, Iivari and Iivari (2006) do not consider user-centered design as a *"separate system development approach,"* as they believe it does not entail aspects of system development either in the requirements phase or in the technical implementation phase. In our survey, 79% of the respondents support the opinion that standardized design techniques can act as a set of criteria against which software may be inspected. Similarly, in all the phases of our empirical analysis, we have found a significant relationship between user-centered design techniques and OSS usability improvement.

Crowston et al. (2003) maintain that the measurement of success and quality in an OSS project is necessary, because millions of users are dependent on OSS systems. Çetin and

Göktürk (2008) do not see usability as the main stimulus behind OSS development; rather, they believe that such development, referred to as *"the freedom of the movement,"* does not necessarily imply usability within software. Consequently, these authors propose a measurement framework for assessing OSS projects, which is required for self-evaluation. In our survey, 69% of the respondents believe that usability assessment is of prime importance, especially considering the distributed environment and the cultural diversity among users. Moreover, our empirical study determined a significant relationship between OSS usability and usability assessment.

Nichols et al. (2001) believe that the less technically-oriented users within the OSS community are challenged in reporting errors and debugging the software. These authors cite examples where users had significant problems with software behavior and documentation even though the developers were pleased with both aspects. Similarly, 90% of the participants in our survey agreed that the proper documentation of OSS projects increases the understandability and learnability of software. Additionally, all phases of our empirical investigation support the positive impact of proper documentation on OSS usability.

5.5.1 Limitations of the Study & Threats to External Validity

Empirical studies are always subject to certain limitations, and although we performed a number of measures to reduce the threats to external validity and increase the reliability, there are still some limitations to this study. First, while there may be other contributing factors that influence OSS usability besides the five independent variables, these factors relate most clearly to the scope of this study, which was to examine the contributing factors from the viewpoint of OSS contributors. Other contributing factors, such as usability bug reporting, interactive help features, usability learning and usability guidelines, usability experts' opinion, incremental design approach, usability testing and knowledge of user-centered design methods are considered in three of our other studies, which are being conducted from the perspective of OSS developers, users and the industry. Another notable limitation of this study involves the small sample size. The smaller sample size, which occurs with the number of projects and respondents, is a potential threat to the external validity of this study. However, the number of participants is limited because few contributors consider usability and its related issues as a top priority.

In this study, we have empirically investigated the effect of key factors on OSS usability and answer the research question stated in this investigation. The empirical results of this study strongly support the hypotheses that users' feedback, design techniques, usability assessment and documentation are positively associated with the usability of an OSS project. However, we could not find any statistical significance for the effect of usability at the architectural level on OSS usability improvement.

Chapter 6

6 An Empirical Study of OSS Usability – The Industrial Perspective

Although user-centered designs are gaining popularity in OSS, usability is still not considered as one of the main objectives in many design scenarios. In this chapter, we analyze the impact of various usability factors, including understandability, learnability, operability and attractiveness, on OSS usability. The research model of this empirical study establishes the relationship between the key usability factors and OSS usability from industrial perspective. In order to conduct the study, a data set of 105 industry users is included. The results of the empirical investigation indicate the significance of the key factors for OSS usability (Raza and Capretz 2010-b).

6.1 Literature Review of Key Factors

In ISO/IEC 9126-1 (2001), understandability is defined as *"the capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use."* According to Seffah and Metzker (2004), software developers and usability experts can both benefit if they understand the culture and practices of HCI and SE and learn techniques for improving the communication between the two disciplines. Furthermore, Mørch et al. (2004) realize that the understandability of end users can be increased if the developers could understand the semantics of integrating different user interface components. In highlighting the diversity among end users, Shneiderman (2000) states that users vary in the amount of time required for understanding and acquiring knowledge about new tools and user interfaces.

ISO/IEC 9126-1 (2001) defines learnability as *"the capability of the software product to enable the user to learn its application."* Seffah et al. (2006) identify the need for more comprehensive guidelines to *"account for the degree of influence of individual quality factors, such as the role of learnability versus understandability in usability problems."* Additionally, Mishra and Hershey (2004) emphasize the background knowledge of users and state that their understanding of requirements can develop improved learning tools.

Yunwen and Kishida (2003) consider learning as one of the main motivational forces that results in the participation of both users and software developers in the OSS culture. Specifically, these authors believe that new users are attracted to OSS because of its high quality whereas developers are attracted to OSS due to its learning opportunities.

Operability is defined in ISO/IEC 9126-1 (2001) as *"the capability of the software product to enable the user to operate and control it."* Henderson (2005) encourages developers to produce software having a usable and operable interface, which could meet user needs and satisfy their value expectations. Ideally, Iivari and Iivari (2006) believe that each individual's needs should be supported by a system; however, they realize that in reality, designers cannot meet the requirements of each user, and, as a result, users should be prepared to make some compromises to have a uniform and compatible system. Accordingly, these authors state that *"in certain situations the prospective users can all participate directly in the process, but in many cases only selected user representatives are involved."*

ISO/IEC 9126-1 (2001) defines attractiveness as *"the capability of the software product to be attractive to the user."* Chrusch (2000) believes that the proper application of usability techniques results in a good user interface. Specifically, he observes that *"many people misinterpret the visual design of an interface as the interface itself, but doing so ignores the entire interaction sequence needed to complete a task."* Juristo (2009) maintains that development teams err when they believe that a system can be made usable merely by utilizing the appropriate font, color, and controls.

6.2 Study Model and Hypothesis

The study model presented in Figure 6.1 is used to analyze and empirically investigate the relationship between the key usability factors and the usability of open source software. The key usability factors studied in this study have been taken from the standard ISO/IEC 9126-1 (2001). Specifically, our aim is to investigate the answer to the following research question:

Research Question: *How do understandability, learnability, operability and attractiveness affect OSS usability from the industrial perspective?*

There are four independent and one dependent variable in this research model. The four independent variables, the usability factors, include Understandability, Learnability, Operability and Attractiveness. On the other hand, the dependent variable of this study is OSS usability. The multiple linear regression equation of the model is as follows:

$$\text{OSS Usability} = \gamma_0 + \gamma_1 v_1 + \gamma_2 v_2 + \gamma_3 v_3 + \gamma_4 v_4 \qquad (6.1)$$

where γ_0, γ_1, γ_2, γ_3 and γ_4 are the coefficients and v_1, v_2, v_3 and v_4 are the four independent variables. In order to empirically investigate the research question, we hypothesize the following:

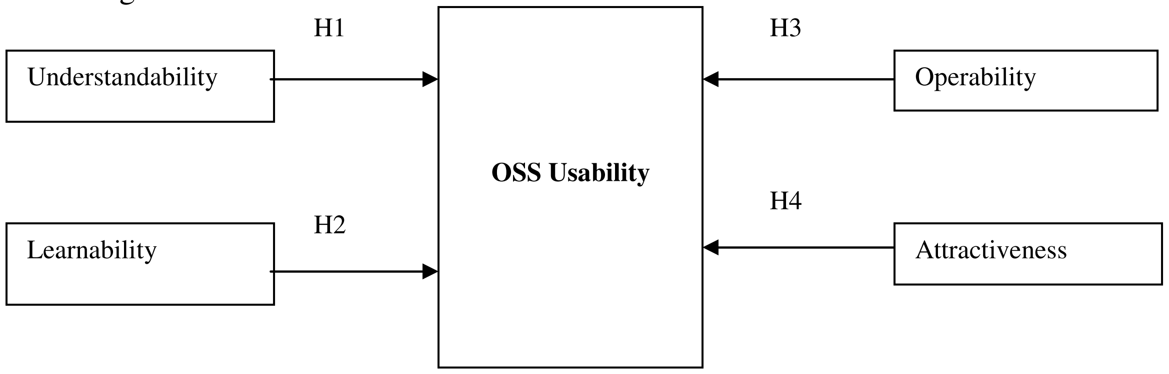


Figure 6.1: Study Model (Industrial Perspective)

The four hypotheses illustrated in the study model are further described in Table 6.1

Table 6.1: Study Model Hypotheses (Industrial Perspective)

Hypothesis #	**Statement**
H1	Understandability is positively related to OSS usability.
H2	Learnability is positively related to OSS usability.
H3	Operability positively affects usability in OSS.
H4	Attractiveness has a positive impact on OSS usability.

6.3 Research Methodology

The research conducted and presented in this chapter includes the empirical results of a survey. In this study, the target population includes multinational companies whose employees are OSS users. Thirty companies consented to participate in this study, with the assurance of confidentiality for both the organization and the individuals. The participating organizations are involved in a wide range of operations, such as pharmaceuticals, telecommunications, automobile manufacturing, information technology and consumer electronics. Specifically, these organizations include North American and European multinational companies, and they vary in size from small to large scale. We requested that the companies in the study distribute the questionnaires within their various departments, so that we have several responses from within the same organization. In particular, we required that the respondents possessed the minimum educational qualification of an undergraduate degree.

The survey was implemented by using the survey tool *"kwiksurveys"*. It was started in the last week of March 2010, and it was closed after three weeks, with 105 responses. We assured the participants that our survey neither required their identity nor would be recorded. However, to support our data analysis of the respondents' experience, we asked them, *"do you agree that applying one of the concepts/techniques expressed by the above key factors, usability will, in your opinion, improve the product you are working on?"* Out of 105 total responses, 81% agreed that in their experience, the application of our key factors will improve the usability of their application; of the remaining participants, 16% were neutral and 3% disagreed with this statement, as reflected in Figure 6.2.

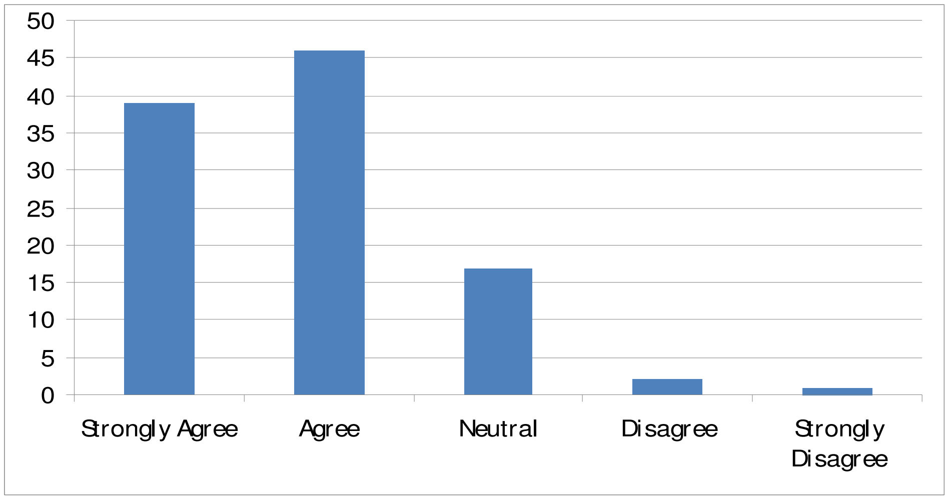


Figure 6.2: Application of Usability Factors in Respondents' Products

6.3.1 Data Collection and the Measuring Instrument

In this study, the questionnaires presented in Tables 6.2 to 6.6 were intended to learn the extent to which these usability factors were important for OSS usability. The questionnaires required the respondents to indicate the extent of their agreement or disagreement with statements on a five-point Likert Scale. We used twelve separate items to measure the independent variables and three items to measure the dependent variable of OSS usability. For all items associated with each variable, the Likert Scale ranged from "Strongly Agree" (1) to "Strongly Dis-agree" (5). There were three items for each independent variable, which were designed to measure the extent to which that particular variable is practiced. The items for all four usability factors are labeled sequentially and are numbered 1 through 12 in Tables 6.2 to 6.5. The items for the independent variable, OSS Usability, are labeled sequentially from 1 through 3, as shown in Table 6.6.

Table 6.2: Understandability Measuring Instrument

Understandability: *"The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use"* (ISO/IEC 9126-1 2001).
1. Consistency in open source applications would increase understandability and hence usability.
2. Software that is easy to understand encourages the user's involvement.
3. Software inconsistency results from an inability to understand users' expectations. (Chrusch 2000)

Table 6.3: Learnability Measuring Instrument

Learnability: *"The capability of the software product to enable the user to learn its application"* (ISO/IEC 9126-1 2001).
4. Learnability increases accessibility and hence usability.
5. In an OSS environment, developers generally compromise learnability for efficient products.
6. OSS developers must understand the limits of their target users to make a system learnable.

Table 6.4: Operability Measuring Instrument

Operability: *"The capability of the software product to enable the user to operate and control it"* (ISO/IEC 9126-1 2001).
7. Learnable software is more operable and hence more usable.
8. Operability is directly proportional to user satisfaction.
9. A modularized system design, where users encounter difficulty levels gradually and progressively, results in operable software. (Yunwen and Kishida 2003).

Table 6.5: Attractiveness Measuring Instrument

Attractiveness: *"The capability of the software product to be attractive to the user"* (ISO/IEC 9126-1 2001).
10. Software that is pleasant to use is more usable.
11. An effective user interface design is the result of properly applied usability techniques and practices.

(Chrusch 2000)
12. Usability is about *“total user experience”* rather than merely an attractive user interface. (Markov 2003)

Table 6.6: OSS Usability Measuring Instrument

Usability: *“The capability of the software product to be understood learned, used and attractive to the user, when used under specified conditions”* (ISO/IEC 9126-1 2001).
1. The adaptation of proven methods in the OSS environment ensures a higher product quality and addresses usability issues. (Hedberg et al. 2007)
2. In order to understand end users’ requirements and expectations, software developers need to enhance their communication with target users instead of relying on their instincts. (Koppelman and Van Dijk 2006).
3. The extent to which a system is useful and easy to use has a major role in determining a user’s intentions. (Te'eni 2007).

6.3.2 Reliability and Validity Analysis of the Measuring Instrument

Two integral features of any empirical study are reliability and validity. While reliability refers to the consistency of a measurement, validity entails the extent to which a measurement reflects its true value. For this empirical investigation, we used the most common approaches for conducting the reliability and validity analysis of the measuring instruments. Specifically, the reliability of the multiple-item measurement scales for the four usability factors was evaluated by using internal-consistency analysis, which was performed using the coefficient alpha (Cronbach 1951). In our analysis, the coefficient alpha ranged from 0.70 to 0.73, as shown in Table 6.7. Nunnally and Bernste (1994) argue that a reliability coefficient of 0.70 or higher is satisfactory. However, van de Ven and Ferry (1980) state that a reliability coefficient of 0.55 or higher is satisfactory, while Osterhof (2001) suggests that a coefficient of 0.60 or higher is satisfactory. Therefore, on the basis of these standards, the variable items developed for this empirical investigation are reliable.

Table 6.7: Coefficient Alpha and Principal Component Analysis of Variables (Industrial Perspective)

Usability Factors	**Item no.**	**Coefficient α**	**PCA Eigen Value**
Understandability	1 - 3	0.73	1.86
Learnability	4 - 6	0.72	1.48
Operability	7 - 9	0.70	1.01
Attractiveness	10 - 12	0.71	1.06

For all four key usability factors, Comrey and Lee's (1992) Principal Component Analysis (PCA) was performed and reported in Table 6.7. Using principal component analysis, we employed the Eigen Value (Kaiser 1970) as a reference point for observing the construct validity. In this study, we used the Eigen Value one-criterion, also known as Kaiser Criterion (Kaiser 1960; Stevens 1986), which indicates that any component having an Eigen Value greater than one is retained. Eigen value analysis revealed that all four variables completely formed a single factor. Therefore, the convergent validity in our study is sufficient for the data.

6.3.3 Data Analysis Procedure

We analyzed the study model and the significance of hypotheses H1 to H4 by using different statistical techniques in three phases. Due to the relatively small sample size, both parametric and non-parametric statistical approaches were used in order to reduce the threats to external validity. In Phase I, we used normal distribution tests and parametric statistics, whereas in Phase II, we used non-parametric statistics. Since our measuring instrument had multiple items for all four independent variables as well as for the dependent variable, their ratings were summed in order to obtain a composite value for each item. For hypotheses H1 to H4, parametric statistical tests were conducted by determining the Pearson correlation coefficient, and non-parametric statistical tests were conducted by determining the Spearman correlation coefficient. In order to manage the limitations of the relatively small sample size and to increase the reliability of the results, the hypotheses of the study model were tested using the PLS technique in Phase III.

According to Fornell and Bookstein (1982) and Joreskog and Wold (1982), the PLS technique is helpful in dealing with issues such as complexity, non-normal distribution, low theoretical information, and small sample size. All statistical calculations were performed using Minitab-15.

6.4 Hypothesis Testing and Results

In the first phase, parametric statistics were used to determine the Pearson correlation coefficient between the individual independent variables, the usability factors, and the dependent variable, OSS usability, as displayed in Table 6.8. Specifically, with a value of 0.42 at $P < 0.05$, the Pearson correlation coefficient between understandability and OSS usability was positive, and hence, hypothesis H1 is justified. Similarly, a Pearson correlation coefficient of 0.42 at $P < 0.05$ was observed between learnability and OSS usability, and hence, this relationship was significant at $P < 0.05$. Hypothesis H3 was accepted based on the Pearson correlation coefficient of 0.51 at $P < 0.05$, which occurred between operability and OSS usability. The positive correlation coefficient of 0.40 at $P < 0.05$ was also observed between OSS usability and attractiveness, which indicated that H4 was also accepted. Hence, all hypotheses were found statistically significant and were accepted.

Non-parametric statistical testing was conducted by examining the Spearman correlation coefficient between the individual independent variables, the usability factors, and the dependent variable, OSS usability, as shown in Table 6.8. The Spearman correlation coefficient between understandability and OSS usability was positive, with a value of 0.40 at $P < 0.05$, and hence, hypothesis H1 is justified. For hypothesis H2, the Spearman correlation coefficient of 0.41 was observed at $P < 0.05$, and thus, a significant relationship was found between learnability and OSS usability. Based on the Spearman correlation coefficient of 0.51 at $P < 0.05$, hypothesis H3, which occurred between Operability and OSS usability, was accepted. A positive Spearman correlation coefficient of 0.37 at $P < 0.05$ was also observed between OSS usability and attractiveness, which indicated that H4 was also accepted. Hence, the hypotheses H1, H2, H3 and H4 were found statistically significant and were accepted based on non-parametric analysis.

Table 6.8: Hypotheses testing using parametric and non-parametric correlation coefficients (Industrial Perspective)

Hypothesis	**Usability Factor**	**Pearson Correlation Coefficient**	**Spearman Correlation Coefficient**
H1	Understandability	0.42*	0.40*
H2	Learnability	0.42*	0.41*
H3	Operability	0.51*	0.51*
H4	Attractiveness	0.40*	0.37*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$.

In the third phase of statistical testing, the PLS technique was used to perform the cross-validation of results obtained in Phases I and II. Specifically, the direction and significance of hypotheses H1 to H4 were examined. In the PLS method, the dependent variable of our study model, OSS usability, was used as the response variable, and the independent key usability factors were used as the predicators. The test results containing observed values of the path coefficient, R^2 and the F-ratio are shown in Table 6.9. Understandability was significant at $P < 0.05$, with a path coefficient of 0.63, an R^2 value of 0.25 and an F-ratio of 27.54. Furthermore, learnability had path coefficient of 0.85, an R^2 value of 0.65 and an F-ratio of 156.44, and thus, it was significant at $P < 0.05$. Operability had the same direction as proposed in hypothesis H3, with path coefficient of 0.80, an R^2 value of 0.56 and an F-ratio of 103.90 at $P < 0.05$. Finally, attractiveness had a path coefficient of 1.08, an R^2 value of 0.61 and an F-ratio of 128.35 at $P < 0.05$, hence concurring with hypothesis H4.

Table 6.9: Hypotheses testing using PLS regression (Industrial Perspective)

Hypothesis	**Usability Factor**	**Path Coefficient**	**R^2**	**F- Ratio**
H1	Understandability	0.63	0.25	27.54*
H2	Learnability	0.85	0.65	156.44*
H3	Operability	0.80	0.56	103.90*
H4	Attractiveness	1. 08	0.61	128.35*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The purpose of study model testing is to provide empirical evidence that our key factors play a significant role in open source software usability. The testing process consists of conducting regression analysis and reporting the values of the model coefficients and their direction of association. The multiple linear regression equation for our study model is depicted in Equation 6.1. In this process, we considered OSS usability as the response variable and the key factors as predicators. The analysis results of the study model are displayed in Table 6.10. The path coefficient of the four variables, understandability, learnability, operability and attractiveness, were positive, and their t-statistics were considered statistically significant at $P < 0.05$.

Table 6.10: Multiple Linear Regression Analysis of the Study Model (Industrial Perspective)

Model coefficient	**Model coefficient**	**Coefficient value**	**t-value**
Understandability	γ_1	0.18	2.73*
Learnability	γ_2	0.15	1.72*
Operability	γ_3	0.15	2.12*
Attractiveness	γ_4	0.24	2.85*
Constant	γ_0	2.72	2.77*

* Significant at $P < 0.05$. ** Insignificant at $P > 0.05$

The R^2 and adjusted R^2 values of the overall study model were significant at $P < 0.05$, being 0.40 and 0.37, with an F-ratio of 14.97.

Recapping Equation 6.1 by inserting model coefficient values, we get:

$$\text{OSS Usability} = 2.72 + 0.18v_1 + 0.15v_2 + 0.15v_3 + 0.24v_4$$

where v_1, v_2, v_3 and v_4 are the four independent variables.

6.5 Discussion of Results

It is generally believed that testing procedures, especially those related to usability, are conducted differently depending on whether they are for closed proprietary software or for OSS projects. However, many testing issues remain common to both types of software, and, as a result, while most of the questions in our survey are specifically related to OSS, some questions are related to testing procedures in general.

For questions related to understandability, we have asked the respondents about the relationship between consistency and understandability. In total, 79% of our respondents agreed that consistency in software design would increase understandability and hence usability, while 16% remained neutral and only 5% disagreed with the statement. The statement that *"software that is easy to understand encourages the user's involvement,"* is equally important for users of both OSS and proprietary software. As a result, 81% of users agreed with this statement, 13% of users remained neutral and 6% of users disagreed. Finally, the statement that *"software inconsistency results from an inability to understand users' expectations"* resulted in 37% of users that agreed, 40% of users that disagreed and 23% of users that remained neutral. Thus, overall, as our statistical analysis indicates that hypothesis H1, which states that *"understandability is positively related to OSS usability*", is found significant and has been accepted in the analysis.

Moreover, we asked our respondents about the relationship between learnability and usability. 83% of the participants agreed that learnability increases accessibility and hence usability, while 14% remained neutral and 3% disagreed. Additionally, we asked the respondents to state whether learnability may be compromised by the efforts of developers in producing efficient products; 34% agreed that this was the case, while 42% remained neutral and 24% disagreed with the statement. Finally, 72% of the participants agreed that in order to make systems learnable, OSS developers must understand the limits of their target users. On the other hand, 14% remained neutral and 14% disagreed with the statement. On the basis of the statistical investigation, hypothesis H2, which states that *"learnability is positively related to OSS usability",* has been accepted.

In order to maintain the unbiased nature of our statements, we asked our respondents whether or not they agree that learnable software is operable and usable. Consequently, 76% of the participants believed that more learnable software is more operable and hence more usable, while 13% remained neutral and the remaining 11% disagreed. For the statement that "*operability is directly proportional to user satisfaction,"* 45% of the respondents agreed, 31% remained neutral and 24% disagreed. Lastly, the belief that *"a modularized system design, where users encounter difficulty levels gradually and progressively, results in operable software"*, was held by 52% of the respondents, while 31% were neutral and 17% disagreed with the statement. Therefore, hypothesis H3, which maintains that, *"operability positively affects usability in OSS*", is supported by the statistical analysis of our survey and thus accepted.

Although our survey statements about attractiveness seem equally applicable to both proprietary and open source software, our respondents, who were from the industry, replied to these statements in the context of their experience with OSS. 67% of the respondents agreed that pleasant software entails usable software, while 21% remained neutral and 12% disagreed Moreover, 79% of the respondents agreed that "*an effective user interface design is the result of properly applied usability techniques and practices.*" On the other hand, 18% of the participants remained neutral and only 3% disagreed. Similarly, 94% of the respondents believed that usability is about "*total user experience*", rather than only about an attractive user interface (Markov, 2003); alternatively, 4% of the respondents remained neutral and 2% disagreed with this statement. Thus, our statistical analysis supports hypothesis, H4, which states that *"attractiveness has a positive impact on OSS usability,"* and hence, it is accepted in the study.

6.5.1 Limitations of the Study & Threats to External Validity

Empirical investigations of software engineering processes and products are performed through surveys, experiments, metrics, case studies and field studies (Singer and Vinson 2002). Kitchenham et al. (2004) recommend that *"software engineering would benefit from adopting what it can of the evidence approach provided that it deals with the specific problems that arise from the nature of software engineering."* Wohlin et al. (2000) identify the ways in which the threats to external validity limit the researcher's ability to generalize the results of his/her experiment to industrial practice. In our study, we needed to support the external validity of our random sampling technique. Accordingly, we made a considerable effort to receive responses from many industry users; however, the total number of respondents was only 105 individuals. Although the proposed approach has some potential to threaten external validity, we followed the appropriate research procedures by conducting and reporting tests to improve the reliability and validity of the study. Furthermore, certain measures were also taken to ensure the study's external validity.

The increasing popularity of empirical methodology in software engineering has also raised concerns with ethical issues (Faden et al 1986; Katz 1972). However, we have followed the recommended ethical principles to ensure that our empirical investigation would not violate any form of recommended experimental ethics.

Another aspect of validity is concerned with whether or not the study reports results that correspond to previous findings. In order to address this concern, we have used four independent variables from ISO/IEC 9126-1 (2001). This standard categorizes software quality attributes into six categories: functionality, reliability, usability, efficiency, maintainability and portability. Furthermore it classifies usability into understandability, learnability, operability and attractiveness.

The results of this study strongly support the hypotheses that understandability, learnability, operability and attractiveness have a positive impact on the usability of OSS projects. Therefore, this empirical investigation provides justification for considering these key factors as measuring instruments for our OSS usability maturity model.

Chapter 7

7 An Open Source Usability Maturity Model (OS-UMM)

In open source projects, usability aspects cannot be improved unless there are ways to test and measure them. Hence, the increasing popularity of open source projects among novice and non-technical users necessitates a usability evaluation methodology. Consequently, this chapter presents a usability maturity model specifically aimed at usability-related issues for open source projects (Raza et al. 2010-c). In particular, the model examines the relationship between open source projects and their usability aspects. The measuring instrument of the model contains factors that have been selected from four of our empirical studies, which examine the perspectives of OSS users, developers, contributors and the industry. In addition to presenting the usability maturity model, this chapter discusses assessment questionnaires, a rating methodology and two case studies.

7.1 Framework of OS-UMM

In comparison to other aspects of software, usability issues are more subjective in nature and hence more controversial. For example, a user interface (UI) element may be clearer to some users than it is to others. However, despite their subjective nature, usability aspects need to be tested and measured objectively. The increasing popularity of usability assessment in open source software projects necessitates a usability maturity evaluation methodology. Consequently, we present a usability assessment methodology for OSS projects, which is termed as the Open Source Usability Maturity Model (OS-UMM)

The assessment questionnaires, which aim to collect information about the usability of an OSS project, comprise of the framework of our methodology. Apart from some of its limitations, the methodology contributes significantly to the area of open source software by addressing the extremely important topic of usability assessment. In four of our empirical studies from OSS users, developers, contributors and industry members, we have been able to identify nineteen key usability factors. Three of these factors, including Usability Guidelines, Usability Expert Opinions and Usability at an Architectural Level, were deemed statistically insignificant for OSS usability. After combining several overlapping factors, such as User Requirements and User Expectations, Incremental Design Approach, Design Techniques and Knowledge of UCD Methods, and Usability Testing and Usability Assessment, we have obtained eleven key usability practices, which are depicted in Table 7.1.

Chrissis et. al. (2006) state that "e*ach maturity level matures an important subset of the organization's processes, preparing it to move to the next maturity level*." The methodology for assessing usability process activities aims to establish a comprehensive strategy for evaluating usability in an OSS project. Specifically, this framework describes the OSS usability assessment methodology and determines the current maturity level of the usability process in an OSS project. Furthermore, it is structured to ascertain the way in which various usability process activities are conducted during the lifecycle of an OSS project. In general, the maturity assessment of OSS usability aims to coordinate open source software with usability-related process activities. The functional framework consists of a set of questionnaires specifically designed to evaluate the usability maturity at each level.

Table 7.1: Configuration of OSS Usability Assessment Methodology

Dimension No.	Dimension	Practice No.	Key Usability Factors
1.	Usability Methodology	1.	Users' Requirements
		2.	Users' Feedback
		3.	Usability Learning
2.	Design Strategy	4.	UCD Methodology
		5.	Understandability
		6.	Learnability
		7.	Operability
		8.	Attractiveness
3.	Assessment	9.	Usability Bug Reporting
		10.	Usability Testing
4.	Documentation	11.	Documentation

As depicted in Table 7.1, the usability assessment methodology for an OSS project includes eleven key usability factors, which are grouped into a set of four dimensions that include "*Usability Methodology", "Design Strategy", "Assessment"* and *"Documentation"*. In particular, the dimension of *"Usability Methodology"* incorporates Users' Requirements, Users' Feedback and Usability Learning. The *"Design Strategy"* dimension covers User-Centered Design Methodology, Understandability, Learnability, Operability and Attractiveness, and the *"Assessment"* Dimension comprises Usability Bug Reporting and Usability Testing.

The usability maturity of an OSS project is determined by the extent to which the project managers and developers agree with each statement in the questionnaire. The number of statements varies for each maturity level as well as for each usability factor. We will use the following set of abbreviations to identify each factor: Users' Requirements (UR), Users' Feedback (UFB), Usability Learning (UL), User-Centered Design Methodology (UCD), Understandability (U), Learnability (L), Operability (O), Attractiveness (Att), Usability Bug Reporting (UBR), Usability Testing (UT) and Documentation (D).

In the questionnaires for our maturity model, the following abbreviations and symbols are used:

UF = Usability Factor

ML = Maturity Level (an integer)

S = Statement

UN = Usability Factor Number (an integer)

Each questionnaire is numbered as S.ML.UF.UN

7.1.1 Level 1: Preliminary

The first level for the usability maturity of open source software projects is referred to as *"Preliminary",* which indicates that the software project does not have a stable and organized methodology for implementing usability. In this level, there is no evidence that the OSS project team practices usability to improve the software quality of their project. Additionally, there are no defined procedures for collecting user requirements and feedback, implementing usability learning, incorporating user design methodology, performing usability assessment and providing documentation. In general, it is evident that usability is not considered to be important, and, consequently, the project does not have sufficient resources and skills to strategically implement usability. A project that is not qualified for any of the levels from Level 2 to Level 5 is, by default, considered to be at Level 1. As a result, our model has no measuring instrument for assessing the usability of an OSS project when it is at the *"Preliminary*" level.

7.1.2 Level 2: Recognized

The next OSS usability level has been defined as *"Recognized."* In this level, project teams recognize the potential benefits of usability in open source projects, and accordingly, they demonstrate interest in usability. Also, teams make efforts to collect user requirements and feedback, and project developers acquire knowledge of UCD methodology. Both teams and developers recognize the fact that end users face difficulties in reporting usability related errors and hence understand the need for usability assessment plans. Additionally, project managers realize the importance of documentation at various stages of software project development. Overall, the project team understands and recognizes the importance of usability in the success of their project, which is in the phase of developing an infrastructure for usability implementation. The measuring instrument for assessing the usability maturity of an OSS project, when it is at the *"Recognized"* level, is illustrated below.

UF.2.1 Users' Requirements

S.2.1.1 Project designers and developers agree that the only way to enhance the software acceptance level is to address users' requirements.

S.2.1.2 The project team is working on a strategy to acquire prospective users' requirements.

UF.2.2 Users' Feedback

S.2.2.1 Project managers collect and analyze feedback from most of the personnel involved in the project.

S.2.2.2 Project management realizes that the project's success is mainly dependent upon users' responses and feedback.

S.2.2.3 There is still a lack of systematic and planned management of users' responses.

UF.2.3 Usability Learning

S.2.3.1 Project designers and developers are acquiring knowledge about the domain of usability engineering.

S.2.3.2 Usability-related issues are learned and managed through the collaboration of viewpoints and open discussion.

S.2.3.3 Plans are established for at least 20% of the developers to have formal training in learning usability design principles.

UF.2.4 UCD Methodology

S.2.4.1 The project team recognizes the importance of the UCD methodology.

S.2.4.2 Project developers consider UCD as an important tool to achieve the desired acceptance level of their product for the target users.

S.2.4.3 At least 20% of the software development team members have background knowledge about UCD methods.

UF.2.5 Understandability

S.2.5.1 Project designers and developers are acquiring knowledge on how to increase the understandability of their product among the users.

S.2.5.2 The OSS team still lacks a systematic strategy for enhancing the understandability level of their software project based on the limitations of their target users.

UF.2.6 Learnability

S.2.6.1 The project team realizes the need for increasing the learnability of their product for their prospective users.

S.2.6.2 There is a lack of technical knowledge among team members about the ways in which to increase the learnability of their software.

S.2.6.3 The project team considers increased learnability as an option for reducing the number of complaints from users.

UF.2.7 Operability

S.2.7.1 Project developers agree that what is operable for them may not be operable for end users.

S.2.7.2 The project team understands the importance of operability and hence realizes the need for increasing the operability of their product.

S.2.7.3 Developers are working on a strategic plan to enhance the operability of their software projects.

UF.2.8 Attractiveness

S.2.8.1 The project team collects information on how to enhance the attractiveness of their product.

S.2.8.2 There is still a lack of defined strategic plans for increasing the project's attractiveness.

S.2.8.3 The project team promotes innovative ideas for making their product pleasing to its users.

UF.2.9 Usability Bug Reporting

S.2.9.1 The development team realizes the difficulties that users experience in reporting usability-related errors.

S.2.9.2 The project development team is working on plans to provide end users with a convenient way to report errors.

UF.2.10 Usability Testing

S.2.10.1 The project team realizes the need for a usability assessment plan before and after the release of their software.

S.2.10.2 It is recognized that usability testing is as necessary as functional testing.

S.2.10.3 The project team collects information on how to ensure the comprehensive assessment of functional and non-functional requirements.

UF.2.11 Documentation

S.2.11.1 The project team understands the importance of documentation; however, it still lacks a systematic and planned strategy for documentation.

S.2.11.2 The project has no formal documentation of users' requirements, design, testing and future improvement plans.

7.1.3 Level 3: Defined

An OSS project at the *"Defined"* level establishes an infrastructure for implementing usability. Specifically, the project team is able to collect and fulfill its users' requirements and expectations. Furthermore, they are able to collect feedback from users by utilizing a planned strategy for improving both software quality and usability. Project managers understand, define and implement user-centered design principles in their product. Through a systemic monitoring structure, team members have the competency to maintain and enhance a project's understandability, learnability, operability and attractiveness. The project team also adopts essential technical skills to provide users with a convenient usability bug reporting facility and to conduct necessary usability testing. The measuring instrument for the usability maturity of an OSS project at Level 3 is illustrated as follows:

UF.3.1 Users' Requirements

S.3.1.1 Identifying target users and collecting their requirements are considered essential processes for achieving a software project's success.

S.3.1.2 A systematic procedure has been defined to collect prospective users' requirements.

UF.3.2 Users' Feedback

S.3.2.1 A planned strategy has been developed to collect users' feedback.

S.3.2.2 Users' feedback is recorded and maintained regularly.

S.3.2.3 Project managers analyze and use feedback to improve quality of the OSS project.

UF.3.3 Usability Learning

S.3.3.1 The project team is committed to acquire knowledge about usability engineering.

S.3.3.2 The project has adequate resources to allocate for usability learning.

S.3.3.3 At least 20% of the project developers have formal training in usability design principles.

UF.3.4 UCD Methodology

S.3.4.1 User-Centered Design is considered essential for the project to satisfy its potential users.

S.3.4.2 User-Centered Design is implemented in the software design of the project.

S.3.4.3 At least 50% of the software development team members have background knowledge of UCD methods.

UF.3.5 Understandability

S.3.5.1 The OSS team has defined a systematic strategy to enhance the understandability level of their software project.

S.3.5.2 Metrics have been defined to measure the understandability level of the project.

UF.3.6 Learnability

S.3.6.1 The project team has adopted a well-designed methodology to enhance the learnability of their software.

S.3.6.2 At least 50% of team developers have sufficient knowledge and technical abilities to improve the learnability of the project.

UF.3.7 Operability

S.3.7.1 Weak areas related to the project's operability are identified and necessary steps are taken for improvement.

S.3.7.2 Considering user limitations, the project team has developed a strategic plan for enhancing the operability of their software project.

UF.3.8 Attractiveness

S.3.8.1 A strategic plan is defined to increase the attractiveness of the project.

S.3.8.2 The project team makes use of innovative ideas for making their product pleasing for its users.

UF.3.9 Usability Bug Reporting

S.3.9.1 At least 30% of the project developers have acquired the technical abilities to provide convenient usability bug reporting facilities for their users.

S.3.9.2 The project team is committed to improving usability bug reporting facilities for their users.

S.3.9.3 The development team records and maintains the difficulties that users experience in reporting usability-related errors.

UF.3.10 Usability Testing

S.3.10.1 The project team has established a usability assessment plan to test the software before its release.

S.3.10.2 At least 25% of the testers have acquired sufficient technical knowledge for assessing the software project's usability.

S.3.10.3 The project managers allocate resources to implement usability testing with actual users.

S.3.10.4 The project has a defined road map for usability testing during every phase of its development.

UF.3.11 Documentation

S.3.11.1 The project team has formal documentation of user requirements, design and testing.

S.3.11.2 Resources have been allocated for project documentation.

S.3.11.3 The project has well-documented future improvement plans.

7.1.4 Level 4: Streamlined

The fourth level for the usability maturity of an OSS project is referred to as *"Streamlined."* In this level, the project team has acquired sufficient resources for meeting its users' requirements. Additionally, they have established a management system for recording users' feedback and taking the necessary steps to address that feedback. New members of the development team are mandated to learn usability principles and guidelines, and project teams consult usability experts regarding the definition and implementation of user-centered design principles in their product. Through a systemic monitoring structure, the understandability, learnability, operability and attractiveness of projects are regularly monitored. Moreover, quantifiable metrics are used to conduct usability assessments, and the documentation of projects is regularly maintained. The following measuring instrument illustrates the usability maturity of an OSS project when it is at the *"Streamlined"* level.

UF.4.1 Users' Requirements

S.4.1.1 The project has adequate resources and skills for acquiring user requirements and is able meet the expectations of its users.

S.4.1.2 A project team sub-unit monitors and ensures that the software design complies with the user requirements.

UF.4.2 Users' Feedback

S.4.2.1 The project team learns from user feedback and avoids repeating previous mistakes.

S.4.2.2 A well-established management system records user feedback regularly and uses it to take the appropriate action with the project.

UF.4.3 Usability Learning

S.4.3.1 Project developers use formal and informal methods for usability learning.

S.4.3.2 Usability knowledge is used to generate new and innovative ideas.

S.4.3.3 Existing developers are encouraged and new developers are required to have formal training in HCI and usability learning.

UF.4.4 UCD Methodology

S.4.4.1 The project team has adequate resources and skills to implement UCD methodology.

S.4.4.2 Statistically collected information indicates that the project has been successful in satisfying its existing users and attracting new ones through the innovative use of UCD methods.

S.4.4.3 The development team members regularly consult HCI and usability experts regarding UCD principles and their implementation.

UF.4.5 Understandability

S.4.5.1 The understandability among users is regularly monitored through user feedback.

S.4.5.2 The understandability level of the project is measured through previously defined metrics.

S.4.5.3 Necessary steps are taken, recorded and monitored to enhance the project's understandability.

UF.4.6 Learnability

S.4.6.1 New users' ability to learn the software is constantly monitored through their response and feedback.

S.4.6.2 The development team ensures that any incorporation of new features in the software is consistent with the previous design.

UF.4.7 Operability

S.4.7.1 The defined strategic plan is implemented to enhance the operability of the software project.

S.4.7.2 A modularized system design is followed in order to increase the project's operability.

S.4.7.3 Metrics have been developed to quantitatively measure the software project's operability level.

UF.4.8 Attractiveness

S.4.8.1 A strategic plan is implemented to increase the attractiveness of the project.

S.4.8.2 The project team regularly monitors the outcome of innovative ideas to make their product pleasing.

S.4.8.3 User feedback regarding the software's attractiveness is regularly collected and maintained.

UF.4.9 Usability Bug Reporting

S.4.9.1 The project development team has developed a service protocol to provide end users with a convenient way for reporting usability bugs and errors.

S.4.9.2 The project team monitors the use of the developed service protocols for effective and easy ways of reporting usability bugs.

UF.4.10 Usability Testing

S.4.10.1 Project testers learn from previous experience and test results to avoid repeating mistakes.

S.4.10.2 The project sub-unit implements the defined road map for usability testing during every phase of its development.

S.4.10.3 A well-established usability assessment system with quantifiable metrics is implemented to regularly perform usability testing.

UF.4.11 Documentation

S.4.11.1 The project documentation has a clear set of guidelines for handling usability-related issues.

S.4.11.2 The project documentation provides a checklist to inspect usability-related issues.

7.1.5 Level 5: Institutionalized

The highest level for the usability maturity of open source software projects is referred to as *"Institutionalized."* An OSS project at this level considers usability as an asset that is necessary for achieving its goals and remaining successful. Specifically, the project has sufficient resources and skills for collecting user feedback and for understanding user expectations. Similarly, the project team establishes a firm commitment to usability learning and UCD methodology, and they continuously make improvements in the areas of understandability, learnability, operability and attractiveness. Furthermore, they constantly improve the usability bug reporting service through innovative methods and ensure the effectiveness of usability assessment by using quantitative metrics. The measuring instrument for the usability maturity of an OSS project at Level 5 is illustrated below.

UF.5.1 Users' Requirements

S.5.1.1 The project successfully responds to user requirements and meets the expectations of its users.

S.5.1.2 The process of incorporating user requirements has become an essential part of the software design.

S.5.1.3 User requirements are regularly reviewed and updated.

UF.5.2 Users' Feedback

S.5.2.1 Regularly collected feedback is used to improve the project quality, especially in terms of usability.

S.5.2.2 The development team follows a schedule to regularly conduct feedback reviews of the updates made in the project.

UF.5.3 Usability Learning

S.5.3.1 The project team is committed to usability learning and to improving knowledge in the areas of HCI and usability.

S.5.3.2 The project developers successfully employ innovative ideas based on their usability knowledge.

UF.5.4 UCD Methodology

S.5.4.1 The project team considers UCD methodology as an important strategic asset.

S.5.4.2 UCD methodology is a prime design methodology in the project.

S.5.4.3 Research and development in UCD methodology is a continuous process in the project.

UF.5.5 Understandability

S.5.5.1 Feedback from users indicates their satisfaction and their ability to conveniently understand the software features.

S.5.5.2 The defined strategy for enhancing the understandability level of the project is regularly reviewed and updated as required.

UF.5.6 Learnability

S.5.6.1 Interactive help is provided for the end users in order to enhance their learnability of the software project.

S.5.6.2 Metrics are used to measure the learnability level for any new features in the software.

UF.5.7 Operability

S.5.7.1 Advanced features are gradually introduced into the software in order to make it more operable and to give the users more control.

S.5.7.2 Defined metrics are used to identify weaknesses, to quantitatively measure and to improve the project's operability.

UF.5.8 Attractiveness

S.5.8.1 A blend of standardized usability techniques and innovative methods are used to make the user interface pleasing and attractive.

S.5.8.2 The project team makes continual efforts to regularly monitor the strategic plan, to ensure that the outcome of the project is attractive and to monitor its improvement.

UF.5.9 Usability Bug Reporting

S.5.9.1 The project team regularly monitors, maintains and improves the existing service protocol for the effective and easy reporting of usability bugs.

S.5.9.2 Project developers are continuously improving the usability bug reporting service.

UF.5.10 Usability Testing

S.5.10.1 The project team experiments with innovative methods to continuously improve the process of usability testing.

S.5.10.2 Quantifiable metrics are used to effectively and objectively measure usability.

S.5.10.3 A usability test management system keeps track of usability tests and uses the results to improve the software quality and the usability of the project.

UF.5.11 Documentation

S.5.11.1 Project documentation is updated on a regular basis to reflect any modifications.

S.5.11.2 A log has been maintained to record user complaints regarding usability-related issues and the corresponding actions taken.

Table 7.2 summarizes the number of items in the assessment questionnaires for each usability factor in all five maturity levels.

Table 7.2: Framework of OS-UMM

Maturity Level	**Usability Factors & Number of Items in Assessment Questionnaire**											
	UR	**UFB**	**UL**	**UCD**	**U**	**L**	**O**	**Att**	**UBR**	**UT**	**D**	**Total**
Preliminary	None											0
Recognized	2	3	3	3	2	3	3	3	2	3	2	29
Defined	2	3	3	3	2	2	2	2	3	4	3	29
Streamlined	2	2	3	3	3	2	3	3	2	3	2	28
Institutionalized	3	2	2	3	2	2	2	2	2	3	2	25

7.2 Performance Scale and Rating Methodology

7.2.1 Performance Scale

The maturity level of an OSS project is determined by its ability to demonstrate the usability factors for that particular level. In order to ascertain a project's maturity level, we have used a five-point scale to rate a project's performance. The rating is a quantitative indication of the extent to which each statement in the questionnaire describes a project, and, more specifically, the way in which a project fulfills the requirements of a specific maturity level. The ordinal ratings used to measure each usability factor, including *"Fulfilled"*, *"Largely Fulfilled"*, *"Partially Fulfilled"*, *"Not Fulfilled"* and *"Not Applicable"*, are depicted in Table 7.3. To increase the flexibility of our methodology, the rating of *"Not Applicable"* has been included in the model. We have structured the performance scales and their thresholds similar to the BOOTSTRAP methodology (Wang and King 2000) in order to maintain the consistency between our usability assessment and the accepted and validated popular scales. However, the linguistic expressions have been slightly modified according to the design of the questionnaires in our model. Overall, our rating methodology allows stakeholders to evaluate the usability maturity of an OSS project based on their level of agreement with the statements. Thus, to this extent, our methodology uses the self-assessment approach.

Table 7.3: Performance Scale

Scale #	Linguistic Expression of Performance Scale		Rating Threshold (%)
	OS-UMM	**BOOTSRAP**	
4	Fulfilled	Completely Satisfied	≥ 80
3	Largely Fulfilled	Largely Satisfied	66.7 – 79.9
2	Partially Fulfilled	Partially Satisfied	33.3 – 66.6
1	Not Fulfilled	Absent / Poor	≤ 33.2
0	Not Applicable	-	-

7.2.2 Rating Methodology

As previously mentioned, we have partially derived the rating methodology from the BOOTSTRAP algorithm (Wang and King 2000). We have used terms, such as Usability Rating (URt), Number of Fulfilled Statements (NF), Passing Threshold (PT) and Usability Maturity Level (UML).

Let URt [i, j] be a rating of the ith usability factor of the jth maturity level. Subsequently, according to the scales defined in Table 7.3, it can be summarized as:

URt [i, j] = 4, if the fulfillment of the condition / statement is at least 80%

= 3, if the fulfillment of the condition / statement is from 66.7 to 79.9%

= 2, if the fulfillment of the condition / statement is from 33.3 to 66.6%

= 1, if the fulfillment of the condition / statement is less than 33.3%

= 0, if the condition / statement is not applicable.

An ith condition/statement at the jth maturity level is considered fulfilled if URt [i, j] ≥ 3 or URt [i, j] is 0. The number of conditions/statements fulfilled at jth maturity level is defined as:

NF [j] = Number of {URt [i, j] | Fulfilled}

= Number of {URt [i, j] | URt [i, j] ≥ 3 or URt [i, j] = 0}

The usability maturity is considered to be achieved if 80% of the conditions or statements in the questionnaire are fulfilled. Thus, if TN [j] is the total number of statements at the jth maturity level, then the passing threshold at the jth maturity level is defined as:

PT [j] = TN [j] * 80%

With the values calculated to the nearest tenth, the passing threshold of 80% at each usability maturity level is illustrated in Table 7.4.

Table 7.4: Rating Threshold for the Open Source Usability Assessment

Usability Maturity Level	**Total Questions**	**Pass Threshold 80%**
Preliminary	0	Not Applicable
Recognized	29	23
Defined	29	23
Streamlined	28	22
Institutionalized	25	20

The Usability Maturity Level (UML) is defined as the highest maturity level at which the number of fulfilled conditions or statements is greater than or equal to the passing threshold PT [j]; hence:

UML = max{ j | NF [j] ≥ PT [j] }

7.3 Reliability and Validity Analysis of Questionnaires

The two integral features of any empirical study are reliability and validity. Reliability refers to the consistency of a measurement, while validity is the extent to which a measurement reflects its true value. In order to conduct a pilot study, we consulted sourceforge.net to select active projects from different categories that had an activity level of 90% or greater. Subsequently, we established contacts with project managers in these projects. In particular, we sent the managers personalized emails describing the scope and objectives of the study. As a result, we received responses from 10 project managers, who provided us with their extent of agreement to each statement in the questionnaire. The projects involved in this pilot study are categorized as Database (2), Desktop Environment (2), Software Development (2), Education (1), Enterprise (1), Games/Entertainment (1) and Networking (1).

The pilot study allows us to analyze the reliability and the construct validity of the questionnaire. For this empirical investigation, we used the most common approaches for conducting reliability and validity analyses of the measuring instruments. The reliability of the multiple-item measurement scales for the five maturity levels was evaluated by using internal-consistency analysis, which is performed with the coefficient alpha (Cronbach 1951). In our analysis, the coefficient alpha ranges from 0.55 to 0.96, as shown in Table 7.5. Nunnally and Bernste (1994) maintain that a reliability coefficient of 0.70 or higher is satisfactory. However, other researchers are more lenient in this regard; van de Ven and Ferry (1980) state that a reliability coefficient of 0.55 or higher is satisfactory, whereas Osterhof (2001) suggests that a coefficient of 0.60 or higher is satisfactory. Our analysis demonstrates that most of the questionnaire items developed for our maturity model satisfy the criteria of Nunnally and Bernste (1994), whereas most of the items with an alpha coefficient of less than 0.70 still meet the criteria of van de Ven and Ferry (1980) and Osterhof (2001). Therefore, based on the criteria for reliability, all the items developed for this empirical investigation are reliable.

Table 7.5: Reliability Analysis of Usability Factors using Coefficient Alpha

Maturity Level	**Usability Factors**										
	UR	**UFB**	**UL**	**UCD**	**U**	**L**	**O**	**Att**	**UBR**	**UT**	**D**
Preliminary	Not Applicable										
Recognized	0.58	0.64	0.72	0.55	0.62	0.55	0.67	0.90	0.94	0.80	0.93
Defined	0.62	0.73	0.76	0.93	0.74	0.67	0.92	0.55	0.75	0.88	0.64
Streamlined	0.81	0.79	0.73	0.76	0.95	0.74	0.69	0.80	0.61	0.86	0.88
Institutionalized	0.80	0.96	0.87	0.89	0.56	0.63	0.58	0.92	0.93	0.75	0.61

Campbell and Fiske (1959) state that convergent validity occurs when scale items correlate and move in the same direction. Principal component analysis (Comrey and Lee 1992) is performed for all eleven usability factors in each maturity level and reported in Table 7.6. Specifically, we have utilized the Eigen Value (Kaiser 1970) as a reference point for observing the construct validity using principal component analysis. In this study, we have used the Eigen Value one-criterion, also known as the Kaiser Criterion (Kaiser 1960; Stevens 1986), where any component having an Eigen Value greater than one is retained. Eigen Value analysis reveals that the items present in the questionnaires completely form a single factor. Therefore, the convergent validity of our study can be regarded as sufficient.

Table 7.6: Convergent Validity of Usability Factors using Eigen Value-1

Maturity Level	Usability Factors										
	UR	**UFB**	**UL**	**UCD**	**U**	**L**	**O**	**Att**	**UBR**	**UT**	**D**
Preliminary	Not Applicable										
Recognized	1.23	1.46	2.05	1.62	1.18	1.75	1.87	2.56	1.89	2.14	1.87
Defined	1.15	2.05	2.23	2.86	1.74	1.51	1.85	1.44	2.03	3.05	1.76
Streamlined	1.80	1.30	1.75	1.67	2.83	1.27	1.89	2.19	1.44	2.36	1.82
Institutionalized	2.22	1.97	1.81	2.57	1.15	1.23	1.33	1.86	1.87	2.13	1.12

7.4 Case Studies

7.4.1 Assessment Methodology

In order to perform the project's usability maturity assessment, we applied our model to two OSS projects. As mentioned on their websites, these projects are conscious of and realize the importance of usability.

- The participants were informed that the assessment was a part of a Ph.D. research study and neither the identity of an individual nor of a project would be disclosed in the resulting Ph.D. thesis or in any subsequent publication.
- The questionnaires designed for OS-UMM are used to measure the usability maturity of each open source software project. The individuals participating in the study were requested to provide their extent of agreement with each statement by using the performance scale ranging from 0 to 4, as illustrated in Table 7.3. Subsequently, the respondents completed the questionnaire by starting at Level 2 and finishing at Level 5.
- The respondents of this study were either project managers or developers. As in the case of other OSS projects, all communication with the respondents was through email and the survey link. The participants took part in the study voluntarily, as they were neither offered nor paid any compensation.

- Both case studies are discussed in the following sections. In order to protect their privacy, they are referred to as Project "A" and Project "B". Both the projects have been selected from sourceforge.net and have activity levels greater than 90%.
- We received multiple responses from each project, and thus, the amount of bias in the sample is limited. A variety of respondents, both project managers and developers from each organization, provided a more accurate overall description of the project. Also, we performed an inter-rater agreement analysis, which provided information regarding the extent of agreement among the raters within each organization; this analysis is described in Section 7.4.4.

7.4.2 Case Study – Project "A"

Project "A" is classified in the category of Desktop Environment and deals with multiple related issues. The project has more than 300 weekly downloads and is recommended by 97% of the users. Furthermore, the project's mailing list has over 300 subscribed users, who discuss user-related issues as well as provide the required help and support. The extent to which Project A corresponds with the statements in the questionnaires of each maturity level is represented in each cell of Table 7.7. According to the rating method discussed in Section 2.2, a statement is considered agreed upon if the performance scale shown in Table 7.3 is either greater than or equal to 3 or 0. From the data presented in Table 7.7, we have thus computed NF, the Number of Fulfilled Statements, for each level. Accordingly, NF is 23 at Level 2, 9 at Level 3, 6 at Level 4 and 5 at Level 5. According to the rating threshold for OS-UMM, NF at Level 2 has a pass threshold of 80%. Consequently, Project A is therefore at the *"Recognized"* maturity level, Level 2.

Table 7.7: Details of Assessment Results of Case Study A

Recognized		Defined		Streamlined		Institutionalized	
Level – 2		Level – 3		Level - 4		Level – 5	
Q. #	Value	Q. #	Value	Q. #	Value	Q. #	Value
2.1.1	4	3.1.1	4	4.1.1	3	5.1.1	3
2.1.2	3	3.1.2	1	4.1.2	1	5.1.2	4
2.2.1	3	3.2.1	1	4.2.1	4	5.1.3	3
2.2.2	4	3.2.2	4	4.2.2	1	5.2.1	2
2.2.3	3	3.2.3	4	4.3.1	1	5.2.2	1
2.3.1	1	3.3.1	1	4.3.2	4	5.3.1	1
2.3.2	4	3.3.2	1	4.3.3	1	5.3.2	2
2.3.3	3	3.3.3	1	4.4.1	1	5.4.1	1
2.4.1	2	3.4.1	1	4.4.2	1	5.4.2	1
2.4.2	3	3.4.2	1	4.4.3	1	5.4.3	1
2.4.3	3	3.4.3	1	4.5.1	2	5.5.1	4
2.5.1	3	3.5.1	1	4.5.2	1	5.5.2	1
2.5.2	3	3.5.2	1	4.5.3	1	5.6.1	1
2.6.1	2	3.6.1	1	4.6.1	1	5.6.2	1
2.6.2	4	3.6.2	1	4.6.2	4	5.7.1	1
2.6.3	3	3.7.1	4	4.7.1	1	5.7.2	1
2.7.1	3	3.7.2	4	4.7.2	1	5.8.1	2

2.7.2	4	3.8.1	1	4.7.3	1	5.8.2	1
2.7.3	3	3.8.2	4	4.8.1	4	5.9.1	1
2.8.1	4	3.9.1	1	4.8.2	4	5.9.2	1
2.8.2	4	3.9.2	1	4.8.3	2	5.10.1	1
2.8.3	4	3.9.3	1	4.9.1	1	5.10.2	1
2.9.1	1	3.10.1	3	4.9.2	1	5.10.3	1
2.9.2	1	3.10.2	3	4.10.1	1	5.11.1	3
2.10.1	1	3.10.3	3	4.10.2	1	5.11.2	1
2.10.2	4	3.10.4	1	4.10.3	1		
2.10.3	4	3.11.1	1	4.11.1	1		
2.11.1	4	3.11.2	2	4.11.2	1		
2.11.2	4	3.11.3	2				

7.4.3 Case Study – Project “B”

Project “B” is a database project that allows data resources to be accessed and integrated with one another. The project has more than 200 weekly downloads and is recommended by 100% of the users. Furthermore, it has a dedicated mailing list for users, with more than 500 subscribed users to provide each other with help and support in a discussion forum.

Table 7.8: Details of Assessment Results of Case Study B

Recognized		**Defined**		**Streamlined**		**Institutionalized**	
Level – 2		**Level – 3**		**Level – 4**		**Level – 5**	
Q. #	**Value**	**Q. #**	**Value**	**Q. #**	**Value**	**Q. #**	**Value**
2.1.1	3	3.1.1	4	4.1.1	2	5.1.1	4
2.1.2	3	3.1.2	4	4.1.2	2	5.1.2	4
2.2.1	4	3.2.1	4	4.2.1	4	5.1.3	4
2.2.2	4	3.2.2	4	4.2.2	4	5.2.1	4
2.2.3	3	3.2.3	4	4.3.1	3	5.2.2	4
2.3.1	3	3.3.1	3	4.3.2	3	5.3.1	3
2.3.2	4	3.3.2	3	4.3.3	2	5.3.2	4
2.3.3	2	3.3.3	2	4.4.1	2	5.4.1	2
2.4.1	3	3.4.1	4	4.4.2	3	5.4.2	2
2.4.2	3	3.4.2	3	4.4.3	1	5.4.3	2
2.4.3	1	3.4.3	2	4.5.1	4	5.5.1	4
2.5.1	3	3.5.1	3	4.5.2	4	5.5.2	4
2.5.2	2	3.5.2	1	4.5.3	4	5.6.1	3
2.6.1	4	3.6.1	4	4.6.1	4	5.6.2	1
2.6.2	2	3.6.2	4	4.6.2	4	5.7.1	4
2.6.3	3	3.7.1	2	4.7.1	3	5.7.2	1
2.7.1	3	3.7.2	2	4.7.2	2	5.8.1	4

2.7.2	3	3.8.1	2	4.7.3	1	5.8.2	4
2.7.3	4	3.8.2	4	4.8.1	3	5.9.1	4
2.8.1	4	3.9.1	4	4.8.2	4	5.9.2	4
2.8.2	3	3.9.2	4	4.8.3	4	5.10.1	4
2.8.3	4	3.9.3	3	4.9.1	4	5.10.2	1
2.9.1	4	3.10.1	4	4.9.2	4	5.10.3	1
2.9.2	4	3.10.2	4	4.10.1	4	5.11.1	3
2.10.1	3	3.10.3	3	4.10.2	2	5.11.2	2
2.10.2	4	3.10.4	2	4.10.3	4		
2.10.3	3	3.11.1	1	4.11.1	1		
2.11.1	3	3.11.2	1	4.11.2	1		
2.11.2	2	3.11.3	1				

The numerical values entered into each cell of Table 7.8 represent the extent to which Project B corresponds to the statements in the questionnaires for each maturity level. Accordingly, NF is 24 at Level 2, 19 at Level 3, 18 at Level 4 and 17 at Level 5. According to the rating threshold for our OS-UMM, at Level 2, NF has a pass threshold of 80%. Based on the rating method discussed in Section 2.2, Project B is also at Level 2, or, the *"Recognized"* level. The assessment results for both case studies are summarized in Table 7.9.

Table 7.9: Summary of Assessment Results of Case Studies

Usability Maturity Level	**Total Questions**	**Pass Threshold 80%**	**Project – A NF**	**Project – B NF**
Preliminary	None	-	-	-
Recognized	29	23	23	24
Defined	29	23	9	19
Streamlined	28	22	6	18
Institutionalized	25	20	5	17

7.4.4 Inter-Rater Agreement Analysis

Since multiple respondents from a single project may have differing opinions about the various usability factors, we have performed an inter-rater agreement analysis, which provides information about the extent of agreement between the raters within one project (El Emam 1999). According to Lee et al. (2001), inter-rater agreement conforms to reproducibility and adheres to the evaluation of identical processes. In order to evaluate inter-rater agreement, the Kendall coefficient of concordance (von Eye and Mun 2005) is often used. However, Cohen's Kappa (Cohen 1960) is preferred in cases where ordinal data is used.

In our study, we have conducted an inter-rater agreement analysis by using both Kendall and Kappa statistics. The total number of respondents, including both project managers and developers from Projects A and B, were 4 and 6 respectively. The Kendall and Kappa statistics for Project A are reported in Table 7.10.

Table 7.10: The Inter-Rater Agreement Analysis of Project A

Usability Maturity Level	**Kendall's Coefficient of Concordance**		**Cohen's Kappa Statistics**		**Fleiss' Kappa Statistics**	
	Coef.	**X^2**	**Coef.**	**Z**	**Coef.**	**Z**
Recognized	0.97	54.39**	0.84	6.79*	0.84	6.63*
Defined	0.98	55.25**	0.89	7.54*	0.89	7.47*
Streamlined	0.93	50.48**	0.64	5.64*	0.64	5.49*
Institutionalized	0.94	45.27**	0.72	6.42*	0.71	5.83*

* $P < 0.001$; ** $P < 0.005$

The values of Kendall's coefficient, Cohen's Kappa and the Fleiss Kappa coefficients can range from 0 to 1, with 0 indicating a complete lack of agreement and 1 indicating perfect agreement (Landis and Koch 1977). For Kendall's coefficient, higher values indicate a stronger association. In Project A, the Kendall coefficients range from 0.93 to 0.98, as shown in Table 7.10. The benchmark for Kappa (El Emam 1999) includes a four-level scale, where < 0.44 is poor, 0.44 – 0.62 is moderate, 0.62 – 0.78 is substantial, and > 0.78 is excellent. For Project A, the Kappa coefficients range from 0.64 to 0.89, and hence, they are classified as substantial.

For Project B, the Kendall coefficients range from 0.95 to 0.99, and the Kappa coefficients range from 0.62 to 0.68, as shown in Table 7.11. Hence, in the case of Project B, these coefficients are also categorized as being substantial.

Table 7.11: The Inter-Rater Agreement Analysis of Project B

Usability Maturity Level	**Kendall's Coefficient of Concordance**		**Cohen's Kappa Statistics**		**Fleiss' Kappa Statistics**	
	Coef.	**X^2**	**Coef.**	**Z**	**Coef.**	**Z**
Recognized	0.95	53.48**	0.66	4.91*	0.65	4.49*
Defined	0.99	55.62**	0.64	6.07*	0.63	5.01*
Streamlined	0.99	53.91**	0.63	6.19	0.62	4.87*
Institutionalized	0.97	47.66**	0.68	5.19	0.68	5.19*

* $P < 0.001$; ** $P < 0.005$

7.5 Discussion

Maturity models in software engineering enable us to obtain comprehensive information about different processes, their related activities and their current maturity levels. Consequently, organizations can use this information to improve their strategic plans and future activities.

In Chapters 3-6, research models were developed in four different empirical studies to establish the relationship between key usability factors from the perspective of different stakeholders and OSS usability. Accordingly, the significant key factors have been used as measuring instruments to present a *Usability Maturity Model for OSS projects:* OS-UMM, which conducts the usability assessment of open source software projects. The structural composition of OS-UMM consists of assessment frameworks from four dimensions, which are based on the perspectives of OSS developers, users and contributors, including architects, designers, developers, testers and users, as well as the industry. The use of the validated key factors makes OS-UMM distinct from the existing models (refer to Chapter 2).

Subsequently, the model has been used to assess the current maturity of an OSS project by defining an assessment methodology and conducting case studies. Specifically, the methodology for evaluating an OSS project's usability maturity profile has become an integral feature of the *Usability Maturity Model*. This model will assist OSS designers/developers in performing usability assessments for their projects in order to enhance their improvement strategies.

7.5.1 Limitations of the Assessment Methodology

- Our open source usability maturity model is questionnaire-based, and hence, it is susceptible to certain limitations. Although our model, which is based on four empirical studies, incorporates five maturity levels and eleven different usability factors, we may have inadvertently omitted other factors that affect usability maturity, such as project size, project category and the role of usability experts.
- While we applied the most commonly-used approaches in our reliability and validity analysis, our measurements were based mainly on the subjective assessments of project managers and developers.
- Since usability is not considered a top priority in the open source software field, we obtained a limited amount of data from open source project developers and managers.
- Our case studies are based on self-assessment, so our assessment techniques did not account for independent assessments. Our rating methodology quantitatively assesses the maturity level of different usability factors and evaluates the overall usability maturity level of an open source project. However, the model contains a lack of explicit guidelines for improving the usability of a project.
- Although we recognize the limitations of our model, we believe that the usability factors in OS-UMM have been validated through empirical investigations and thus provide a comprehensive approach and a firm foundation for future research in this area.

Chapter 8

8 Conclusions and Future Work

The experiences of end users have a profound impact on the success of a software project, and, as a result, usability and its related issues are a key area of research in open source software. A few of the challenging options for improving OSS usability include understanding users' expectations through empirical investigations, adapting approaches in OSS designs to improve usability and quantifying usability metrics. However, in order to identify the precise areas where improvement is necessary, assessment needs to be performed. Due to a continual increase in the number of both technical and non-technical users, the evaluation of OSS usability requires a comprehensive strategy, which has not yet been fully explored.

Apart from its limitations, this thesis contributes towards establishing a comprehensive strategy for the usability maturity of OSS projects, and it addresses the important issue of usability assessment in open source projects. Our proposed usability maturity model for open source projects is based on eleven key factors that we have identified and empirically analyzed in four previous studies. The framework of the model is comprised of assessment questionnaires, a performance scale, and a rating methodology. Additionally, two case studies have been discussed, where we have examined the performance of two OSS projects in the area of usability.

8.1 Major Contributions

This thesis aims to increase the available research in the area of OSS usability assessment methodologies and provide a viable solution in the form of *A Usability Maturity Model for Open Source Software.* The research conducted and reported in this thesis provides a comprehensive strategy for conducting a usability maturity assessment of an OSS project. The main contributions of the research in this thesis are:

- An empirical investigation to discover certain factors that have an impact on the usability of an OSS project. This study, which is reported in Chapter 3 of the thesis, identifies factors from the perspective of OSS users. Specifically, the study identifies, investigates and validates these key factors and demonstrates the relationship between the factors and OSS usability.

- A study that identifies certain factors with an impact on OSS usability from the perspective of OSS developers. In order to demonstrate the relationship between the key factors and OSS usability, an empirical investigation and validation through a research model has been performed and presented in Chapter 4.
- A recognition of factors affecting OSS usability from the point of view of OSS contributors. An empirical investigation investigates and validates the key factors and demonstrates the relationship between these factors and OSS usability using a study model in Chapter 5.
- An empirical study that identifies certain factors with an impact on OSS usability from an industry perspective. The study establishes the relationship between the identified key usability factors and OSS usability through a research model in Chapter 6.
- A framework for *A Usability Maturity Model for Open Source Software.* This model utilizes the empirically validated key factors as a measuring instrument for OSS projects. In particular, five maturity levels have been used to represent the usability maturity of an OSS project. Additionally, this chapter, Chapter 7, discusses assessment questionnaires, a rating methodology and two case studies.

8.2 Conclusions

The usability assessment of open source projects is an area where relatively little attention has been paid by researchers, and, accordingly, the main contribution of this work is a methodology that evaluates the usability maturity of an OSS project. In the second section of Chapter 1, seven research questions were formulated. The framework of the thesis was guided by the goal of finding answers to those research questions, thus providing a comprehensive and unified methodology for assessing the usability maturity of an OSS project. The research conducted and reported in this thesis provided answers to those seven research questions, as presented below:

RQ-1: What are the key factors that influence the usability of open source software projects from the perspective of users?

Answer: The empirical investigation conducted and reported in Chapter 3 of this thesis confirmed that various usability factors, such as Users' Expectations, Usability Bug Reporting and Fixing, Interactive Help Features and Usability Learning, have a positive impact on the usability of an OSS project. Empirical results of the study indicated that the factors in our research model assist in understanding and analyzing the end users' perception of OSS usability.

RQ-2: Can we identify the key usability factors affecting open source software projects from the viewpoint of OSS developers?

Answer: We identified and examined the effect of key factors on OSS usability from the viewpoint of OSS developers. The empirical study conducted and reported in Chapter 4 of this thesis found that many usability factors, such as Users' Requirements, Incremental Design Approach, Usability Testing and Knowledge of UCD Methods, have a positive effect in promoting OSS usability.

RQ-3: From the perspective of OSS contributors, what are the key factors that affect OSS usability?

Answer: In Chapter 5, a research model has been presented to investigate and establish a relationship between the key usability factors from the perspective of contributors and OSS usability. Empirical results of the study strongly supported the hypotheses that Users' Feedback, Design Techniques, Usability Assessment and Documentation are positively associated with the usability of OSS projects.

RQ-4: What are the key factors that contribute towards the maturity assessment of OSS usability from the perspective of the industry?

Answer: In Chapter 6, an empirical analysis examined the impact of key factors on OSS usability and found that understandability, learnability, operability and attractiveness contribute towards the maturity assessment of OSS usability from the industry's perspective.

RQ-5: Are we able to develop a methodology to assess the usability maturity of an open source software project?

Answer: In Chapter 7, this thesis presented a usability maturity model, OS-UMM, which is intended to assess the usability maturity of an open source software project.

RQ-6: Can specific scales represent the extent of usability maturity in an OSS project?

Answer: This thesis defined five maturity levels to represent the extent of OS-usability maturity in an open source software project (Chapter 7).

RQ-7: Can we develop a methodology for evaluating the project's usability maturity level once the assessment results are manifested?

Answer: In Chapter 7, this thesis presented five performance scales and a rating methodology for assessing the usability maturity of an open source software project.

Due to the paucity of research in this extremely important area, the primary objective of this thesis was to propose a *Usability Maturity Model for Open Source Software Projects.* This thesis identified three research problems in Chapter 1, Section 3 and has provided solutions to those problems.

8.3 Future Work

This study has been primarily focused on two objectives: to identify certain usability factors that may help in improving OSS usability from the perspective of different stakeholders, and to propose OS-UMM, a usability maturity model for OSS projects. Some of the leading research areas and suggested future work in those areas are presented as follows:

- We have used self assessment method to perform case studies. We are planning to enhance the assessment methodology by introducing on-site assessment by identifying documents to review, interview questions and mapping replies to the measuring instrument of our proposed maturity assessment model.

- Instead of developing a more generic model, we could possibly tailor the existing usability models, including OS-UMM, to use them in specific application domains, such as communications, medicine and the automobile industry. Exploratory research in this direction will help in developing more domain oriented models.
- Presently there is no definition of how the improvement plans will be generated and implemented after the assessment. Furthermore, a guideline, regarding how to move up a ladder from one maturity level to another, is missing. We would like to work on these issues as well.
- Regarding the factors that have not been validated in our empirical studies, further studies may be needed to establish whether these factors are relevant or not in the assessment of OSS usability maturity.

References

Aberdour, M. 2007. Achieving quality in open-source software, IEEE Software, 24(1), 58-64.

Abran, A., Surya, W., Khelifi, A., Rilling, J., Seffah, A. and Robert, F. 2003. Consolidating the ISO usability models, 11th Annual International Software Quality Management Conference.

Andreasen, M. S., Nielsen, H.V., Schrøder, S.O. and Stage J. 2006. Usability in open source software development: Opinions and practice, Information Technology and Control, 35A (3), 303 – 312.

Andreasen, M.S., Nielsen, H.V., Schrøder, S.O., and Stage, J. 2007. What happened to remote usability testing? An empirical study of three methods, The SIGCHI Conference on Human Factors in Computing Systems.

Benson, C., Muller-Prove, M. and Mzourek, J. 2004. Professional usability in open source projects: GNOME, OpenOffice.org, NetBeans, CHI '04 extended abstracts on Human Factors in Computing Systems, Vienna, Austria.

Bevan, N. 2006. Practical issues in usability measurement, Interactions, 13(6), 42-43.

Bevan, N. 2008. Reducing risk through human centred design, I-USED, Pisa.

Bevan, N. 2009. International standards for usability should be more widely used, Journal of Usability Studies, 4(3), 106-113.

Bodker, M., Nielsen, L. and Orngreen, R.N. 2007. Enabling user centered design processes in open source communities, Usability and Internationalization, HCI and Culture, 2nd International Conference on Usability and Internationalization, UI-HCII 2007. Held as Part of HCI International 2007, Part I, 10-18.

Bouktif, S., Antoniol, G., Merlo, E. and Neteler, M. 2006. A feedback based quality assessment to support open source software evolution: the GRASS case study, 22nd IEEE International Conference on Software Maintenance, 155-165.

Campbell, D.T. and Fiske, D.W. 1959. Convergent and discriminant validation by the multi-trait multi-method matrix, Psychological Bulletin, 56(2), 81–105.

Capra, E., Francalanci, C. and Merlo, F. 2008. An Empirical Study on the Relationship Between Software Design Quality, Development Effort and Governance in Open Source Projects, IEEE Transactions on Software Engineering, 34(6), 765 – 782.

Çetin, G. and Göktürk, M. 2007. Usability in open source community, ACM Interactions, 14(6), 38-40.

Çetin, G. and Göktürk, M. 2008. A measurement based framework for assessment of usability-centricness of open source software projects, IEEE International Conference on Signal Image Technology and Internet Based Systems, SITIS '08.

Çetin, G., Verzulli, D. and Frings, S. 2007. An analysis of involvement of HCI experts in distributed software development: practical issues - Online communities and social computing, 2nd International Conference, OCSC 2007, held as Part of HCI International 2007, 32-40.

Chrissis M.B., Konran M. and Shrum S. 2006. CMMI: Guidelines for process integration and product improvement, 2nd Edition Addison-Wesley Publishing Company, 9-69.

Chrusch, M. 2000. The whiteboard: seven great myths of usability, Interactions, 7(5), 13 – 16.

Cohen, J. 1960. A coefficient of agreement for nominal scales, Educational and Psychological Measurement, 20, 37–46.

Comrey, A.L. and Lee, H.B. 1992. A First Course on Factor Analysis, 2nd Edition Hillsdale.

Craven, J. and Booth H. 2006. Putting awareness into practice: practical steps for conducting usability tests, Library Review, 55(3), 179-194.

Cronbach, L. J. 1951. Coefficient alpha and the internal consistency of tests, Psychometrica, 16, 297–334.

Crowston, K., Annabi, H. and Howison, J. 2003. Defining open source software project success, 24th International Conference on Information Systems (ICIS), Seattle, WA.

Earthy, J. 1999. Usability maturity model: Processes, Version 2.2, Lloyd's Register, London.

Earthy, J. 1998. Usability maturity model: Human-centredness scale, IE2016 INUSE Deliverable D5.1.4s.

Easterbrooks, S., Singer, J., Storey, M.A. and Damian, D. 2007. Selecting empirical methods for software engineering research, International Conference on Automated Software Engineering, Atlanta, Georgia, USA, 2007.

El Emam, K. 1999. Benchmarking kappa: Inter-rater agreement in software process assessments, Empirical Software Engineering, 4(2), 113–133.

Faden, R.R., Beauchamp, T.L. and King, N.M.P. 1986. A History and Theory of Informed Consent, Oxford University Press.

Faulkner, X. and Culwin, F. 2000. Integrating HCI and SE, ACM SIGCSE Bulletin, 32(3), 61-64.

Feller, J. and Fitzgerald, B. 2000. A framework analysis of the open source software development paradigm, 21st Annual International Conference on Information Systems, Brisbane, Queensland, Australia, 58–69.

Fitzgerald, B. 2006. The transformation of open source software, MIS Quarterly, 30(3), 587-598.

Folmer, E. and Bosch, J. 2004. Architecting for usability: A survey, The Journal of Systems and Software, 70, 61–78.

Fornell, C. and Bookstein, F.L. 1982. Two structural equation models: LISREL and PLS applied to consumer exit voice theory, Journal of Marketing Research, 19, 440–452.

Golden, E., John, B.E. and Bass, L. 2005. The value of a usability-supporting architectural pattern in software architecture design: A controlled experiment, 27th International Conference on Software Engineering, St. Louis, MO, USA.

Granollers T., Lorés J. and Perdrix F. 2003. Usability engineering process model. Integration with software engineering, HCI-Intl'03, Crete-Greece.

Hansen, M., Köhntopp, K. and Pfitzmann, A., 2002. The open source approach — opportunities and limitations with respect to security and privacy, Computers and Security, 21(5), 461-471.

Hedberg, H., Iivari, N. Rajanen M. and Harjumaa L. 2007. Assuring quality and usability in open source software development, 1st International Workshop on Emerging Trends in FLOSS Research and Development, FLOSS, IEEE Computer Society, Washington, DC, May 20 - 26.

Henderson, A. 2005. The innovation pipeline: design collaborations between research and development, Interactions, 12(1), 24 – 29.

Holzinger, A. 2005. Usability engineering for software developers, Communications of the ACM, 48(1), 71-74.

Holzinger, A., Sammer P., and Hofmann-Wellenhof, R. 2006. Mobile computing in medicine: Designing mobile questionnaires for elderly and partially sighted people, Computers Helping People with Special Needs, ICCHP 2006, Lecture Notes in Computer Science LNCS, 4061. Berlin, Heidelberg, New York, Springer, 732-739.

Hornbæk, K. 2006. Current practice in measuring usability: Challenges to usability studies and research, Int. J. Human-Computer Studies, 64, 79–102.

http://www.opensource.org/docs/osd, 2010.

Humphrey, W.S. 1990. Managing the software process, Addison-Wesley.

Iivari, J., and Iivari, N. 2006. Varieties of user-centeredness, 39th Annual Hawaii International Conference on System Sciences - Volume 08 HICSS. IEEE Computer Society, Washington, DC.

Iivari, N. 2009-a. Empowering the users? A critical textual analysis of the role of users in open source software development, AI Soc. 23(4), 511-528.

Iivari, N. 2009-b. Constructing the users in open source software development: an interpretive case study of user participation, Information Technology & People, 22(2), 132-56.

Iivari, N., Hedberg, H. and Kirves, T. 2008. Usability in company open source software context - Initial findings from an empirical case study, Open Source Development, Communities and Quality in IFIP International Federation for Information Processing, 275, 359–365.

International Standard ISO/IEC 9126-1 (2001) Software Engineering – Product Quality – Part 1: Quality model (1st edition, 2001-06-15), 9-10.

ISO 9241. 1997. Ergonomics Requirements for Office with Visual Display Terminals (VDTs).

Joreskog, K. and Wold, H. 1982. Systems under Indirect Observation: Causality, Structure and Prediction, North Holland, The Netherlands.

Juristo, N. 2009. Impact of usability on software requirements and design, Lecture Notes in Computer Science, 5413, Springer-Verlag, 55-77.

Kaiser, H.F. 1960. The application of electronic computers to factor analysis, Educational and Psychological Measurement, 20, 141–151.

Kaiser, H.F. 1970. A second generation little jiffy, Psychometrika, 35, 401–417.

Kaner,C. 2006. Exploratory Testing, Quality Assurance Institute Worldwide Annual Software Testing Conference, Orlando, FL.

Katz, J. 1972. Experimentation with Human Beings. New York: Russell Sage Foundation.

Kitchenham, B.A., Dybå, T. and Jørgensen M. 2004. Evidence-based software engineering, 26th IEEE international conference on Software Engineering, 273-281.

Koch, S. and Neumann, C. 2008. Exploring the effects of process characteristics on product quality in open source software development, Journal of Database Management, 19(2), 31-57.

Koppelman, H. and Van Dijk, B. 2006. Creating a realistic context for team projects in HCI, SIGCSE Bulletin, 38(3), 58-62.

Landis, J., and Koch, G.G. 1977. The measurement of observer agreement for categorical data, Biometrics, 33, 159–174.

Laplante, P., Gold, A. and Costello, T. 2007. Open source software: Is it worth converting? IT Professional, 9(4), 28-33.

Lee, H.Y., Jung, H.W., Chung, C.S., Lee, J.M., Lee K.W., and Jeong, H.J. 2001. Analysis of inter-rater agreement in ISO/IEC 15504-based software process assessment, 2nd Asia– Pacific Conference on Quality Software, 341–348.

Lee, S.Y.T., Kim, H.W., and Gupta, S. 2009. Measuring open source software success, Omega, 37(2), 426-438.

Lethbridge T. 2007. UUMM: User and Usability Maturity Model, personal notes, School of Information Technology and Engineering, University of Ottawa, Personal Information.

Levesque, M. 2005. Fundamental issues with open source software development. (Originally published in Volume 9, Number 4, April 2004), First Monday [Online], 0(0).

Mackie C. 1997. Process excellence and capability determination, BT technology journal 15(3).

Markov, N. 2003. An introduction to the UCD methodology in the current environment, CASCON Workshop Report.

Michlmayr, M., Hunt, F. and Probert, D. 2005. Quality practices and problems in free software projects, 1st International Conference on Open Source Systems, Genova, 11th-15th July 2005, Marco Scotto and Giancarlo Succi (Eds.), 24-28.

Miller, J. 2006. Usability testing: A journey, not a destination, Internet Computing, IEEE. 10(6), 80-83.

Mishra, P. and Hershey, K.A. 2004. Etiquette and the design of educational technology, Communications of the ACM, 47(4), 45-49.

Mørch, A.I., Stevens, G., Won, M., Klann, M., Dittrich, Y. and Wulf, V. 2004. Component-based technologies for end-user development, Communications of ACM, 47(9), 59-62.

Nakagawa, E.Y., de Sousa, E.P.M., de Brito, M.K., de Faria, A.G., Morelli, L.B. and Maldonado, J.C. 2008. Software architecture relevance in open source software evolution: A case study, IEEE 32nd International Computer Software and Applications Conference (COMPSAC), 1234-1239.

Nichols, D.M. and Twidale M.B. 2005. The usability of open source software, First Monday, 8(1). http://firstmonday.org/issues/issue8 1/nichols/

Nichols, D.M. and Twidale, M.B. 2006. Usability processes in open source projects, Software Process: Improvement and Practice, 11(2), 149- 162.

Nichols, D.M., Thomson, K. and Yeates, S.A. 2001. Usability and open source software development, Symposium on Computer Human Interaction. Palmerston North, New Zealand: SIGCHI New Zealand, 49-54.

Nunnally, J.C. and Bernste, I.A. 1994. Psychometric Theory, 3rd ed. McGraw Hill, New York.

O'Reilly, T. 1999. Lessons from open-source software development, Communications of ACM, 42(4), 32–37.

Osterhof, A. 2001. Classroom Applications of Educational Measurement, Prentice Hall, NJ.

Otte, T., Moreton, R. and Knoell, H.D. 2008. Applied quality assurance methods under the open source development model, IEEE 32nd International Computer Software and Applications Conference (COMPSAC), 1247-52.

Paulk, M.C., Weber, C.V., Curtis, B. and Chrissis, M.B. 1995. The Capability Maturity Model: Guidelines for Improving the Software Process/CMU/SEI, Addison-Wesley Publishing Company, 15-28.

Paulson, J.W., Succi, G. and Eberlein, A. 2004. An empirical study of open-source and closed-source software products, IEEE Transactions on Software Engineering, 30(4), 246-56.

Pemberton, S. 2004. Scratching someone else's itch: Why open source can't do usability, Interactions, 11(1), 72.

Polancic, G., Horvat, R.V. and Rozman, T. 2004. Comparative assessment of open source software using easy accessible data, 26th International Conference on Information Technology Interfaces - ITI 2004 (IEEE Cat. No.04EX794), 1, 673-8.

Porter, A., Yilmaz, C., Memon, A.M., Krishna, A.S., Schmidt, D.C. and Gokhale, A. 2006. Techniques and processes for improving the quality and performance of open-source software, Software Process Improvement and Practice, 11(2), 163-76.

Raymond, E.S. 1999. The Cathedral and the Bazaar, O'Reilly, Sebastopol, CA.

Raymond, E.S. 2001. The Cathedral and the Bazaar, Revised Edition, O'Reilly, Sebastopol, CA.

Raza, A. and Capretz, L.F. 2010-a. Contributors' preference in open source software usability: An empirical study, International Journal of Software Engineering & Applications (IJSEA), 1(2), 45 – 64.

Raza, A. and Capretz, L.F. 2010-b. An empirical study of open source software usability – The industrial perspective, International Journal of Open Source Software and Processes, Under Review

Raza, A., Capretz, L.F. and Ahmed, F. 2010-a. Users' perception of open source usability:An empirical study, Engineering with Computers, Under Review

Raza, A., Capretz, L.F. and Ahmed, F. 2010-b. Improvement of Open Source Software Usability: An Empirical Evaluation from Developers Perspective, Advances in Software Engineering, vol. 2010, Article ID 517532, 12 pages, 2010. doi:10.1155/2010/517532

Raza, A., Capretz, L.F. and Ahmed, F. 2010-c. An open source usability maturity model (OS-UMM), Behaviour & Information Technology, Under Review

Rosson, M.B., Carroll J.M. and Rodi C.M.2004.Teaching usability engineering, ACM SIGCSE Bulletin, 36(1), 36-40.

Rusu,C., Rusu ,V. and Roncagliolo, S. 2008. Usability Practice: The appealing way to HCI, 1st International Conference on Advances in Computer-Human Interaction, achi-2008, 265-270.

Sampson, F. 2007. Who said "usability is free"? Interactions, 14(4), 10-11.

Seffah, A. and Metzker, E. 2004. The obstacles and myths of usability and software engineering, Communications of the ACM, 47(12), 71-76.

Seffah, A., Donyaee, M., Kline, R. and Padda, H. 2006. Usability measurement and metrics: A consolidated model, Software Quality Journal, 14(2), 159-178.

Shneiderman, B. 2000. Universal usability, Communications of the ACM, 43(5), 84-91.

Singer, J. and Vinson, N.G. 2002. Ethical issues in empirical studies of software engineering, IEEE Transactions on Software Engineering, 28(12), 1171-1180.

Stamelos, L., Angelis, L., Oikonomou, A. and Bleris, G.L. 2002. Code quality analysis in open source software development, Information Systems Journal, 12(1), 43-60.

Stevens, J. 1986. Applied Multivariate Statistics for the Social Sciences, Hillsdale, NJ.

Te'eni, D. 2007. HCI is in business---focusing on organizational tasks and management, Interactions, 14(4), 16 – 19.

Terry, M., Kay, M. and Lafreniere. B. 2010. Perceptions and practices of usability in the free/open source software (FOSS) community. 28th International Conference on Human Factors in Computing Systems CHI' 10, 999 -1008.

Twidale, M. 2005. Silver bullet or fool's gold: supporting usability in open source software development, 27th International Conference on Software Engineering, 35.

Twidale, M.B. and Nichols, D.M. 2005. Exploring usability discussions in open source development, 38th Annual Hawaii International Conference on System Sciences, 198-208.

van de Ven, A.H. and Ferry, D.L. 1980. Measuring and Assessing Organizations, John Wiley & Son, NY.

Viorres, N., Xenofon, P., Stavrakis, M., Vlachogiannis, E., Koutsabasis, P. and Darzentas, J. 2007. Major HCI challenges for open source software adoption and development, Schuler, Douglas (ed.) OCSC 2007 - Online Communities and Social Computing, 2nd International Conference, Beijing, China. 455-464, Jul. 22-27.

von Eye, A., and Mun., E.Y. 2005. Analyzing Rater Agreement Manifest Variable Methods, London: LEA Publishers.

von Krogh and Spaeth, S. 2007. The open source software phenomenon: characteristics that promote research, Journal of Strategic Information Systems, 16(3), 236-53.

Wikipedia-a, http://en.wikipedia.org/wiki/User_(computing) , 2010.

Wikipedia-b, http://en.wikipedia.org/wiki/User_expectations), 2010.

Wikipedia-c, http://en.wikipedia.org/wiki/Interactive, 2010.

Wikipedia-d, http://en.wikipedia.org/wiki/Learning, 2010.

Wikipedia-e, http://en.wikipedia.org/wiki/Guidelines, 2010.

Wikipedia-f, http://en.wikipedia.org/wiki/P-value, 2010.

Wikipedia-g, http://en.wikipedia.org/wiki/Developer_(software), 2010.

Wikipedia-h, http://en.wikipedia.org/wiki/Perspective_(cognitive), 2010.

Wikipedia-i, http://en.wikipedia.org/wiki/Requirement, 2010.

Wikipedia-j, http://en.wikipedia.org/wiki/Expert, 2010.

Wikipedia-k, http://en.wikipedia.org/wiki/Incremental_and_iterative_development, 2010.

Wikipedia-l, http://en.wikipedia.org/wiki/User-centered_design, 2010.

Wikipedia-m, http://en.wikipedia.org/wiki/Feedback, 2010.

Wikipedia-n, http://en.wikipedia.org/wiki/Software_architecture, 2010.

Wikipedia-o, http://en.wikipedia.org/wiki/Usability#Designing_for_Usability, 2010.

Wikipedia-p, http://en.wikipedia.org/wiki/Software_documentation, 2010.

Wang, Y. and King, G. 2000. Software engineering processes: Principles and Applications, CRC Press, NY. 191 – 219.

Winter, S., Wagner S., and Deissenboeck F. 2008. A comprehensive model of usability, J. Gulliksen et al. (Eds.): EIS 2007, LNCS 4940, 106-122.

Wohlin, C., Runeson, P., Host, M., Ohlsson, M.C., Regnell, B. and Wesslen, A. 2000. Experimentation in Software Engineering, Kluwer Academic Publishers, Norwell, MA.

Yunwen, Y. and Kishida, K. 2003. Toward an understanding of the motivation of open source software developers, 25th International Conference on Software Engineering IEEE Computer Society, Washington, DC, 419-429.

Zhao, L. and Deek F.P. 2006. Exploratory inspection – A learning model for improving open source software usability, Conference on Human Factors in Computing Systems CHI '06, 1589 -1594.

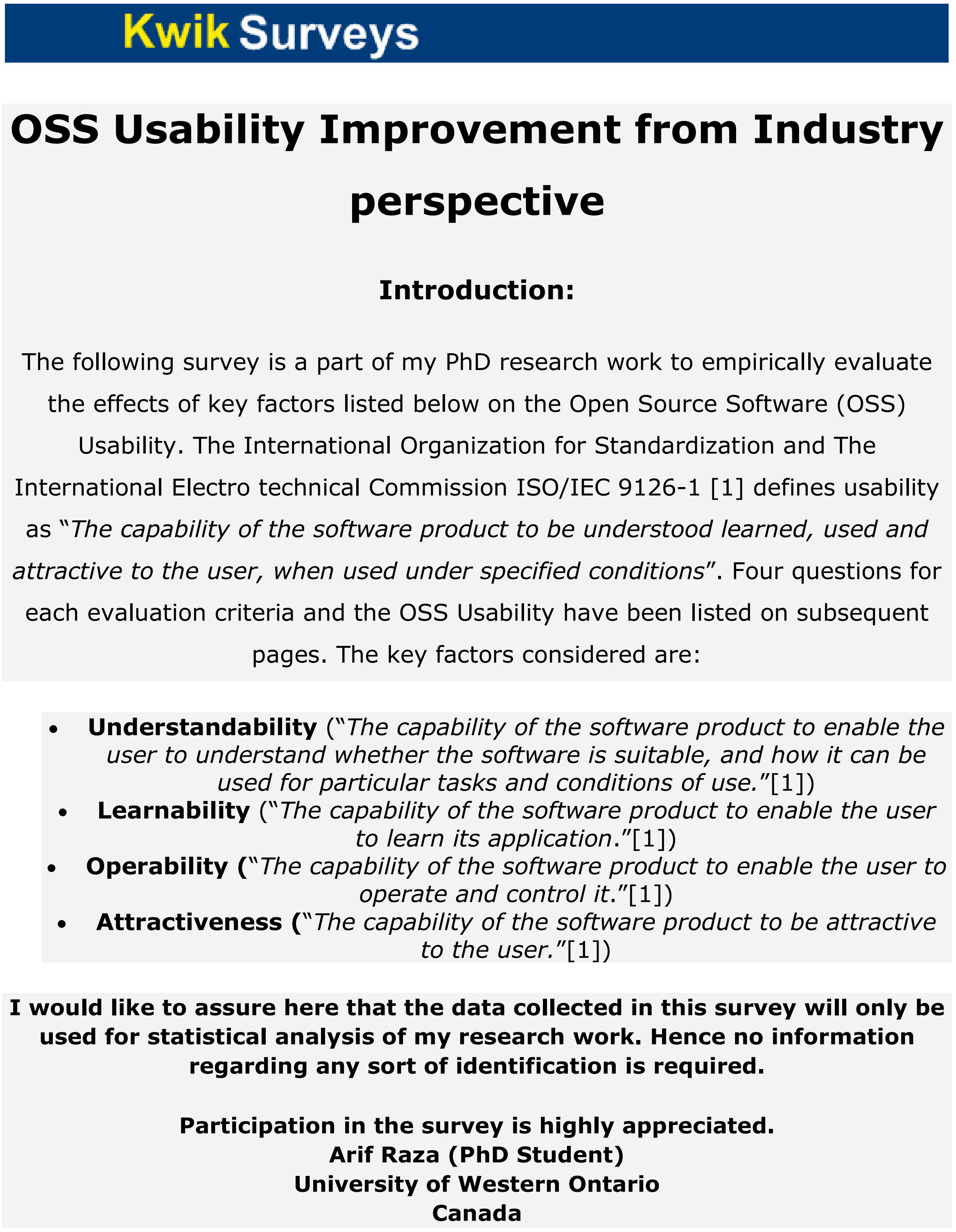
Zhao, L. and Deek, F. 2005. Improving open source software usability, 11th Americas Conference on Information Systems, August 11-14, Omaha, USA, 923-928.

Zhao, L. and Elbaum, S. 2000. A survey on quality related activities in open source, ACM SIGSOFT Software Engineering Notes, 25(3), 54-57.

Zhao, L. and Elbaum, S. 2003. Quality assurance under the open source development model, The Journal of Systems and Software, 66, 65-75.

Appendix A

Screen Shot of Empirical Study Survey



OSS Usability Improvement from Industry perspective

Introduction:

The following survey is a part of my PhD research work to empirically evaluate the effects of key factors listed below on the Open Source Software (OSS) Usability. The International Organization for Standardization and The International Electro technical Commission ISO/IEC 9126-1 [1] defines usability as "*The capability of the software product to be understood learned, used and attractive to the user, when used under specified conditions*". Four questions for each evaluation criteria and the OSS Usability have been listed on subsequent pages. The key factors considered are:

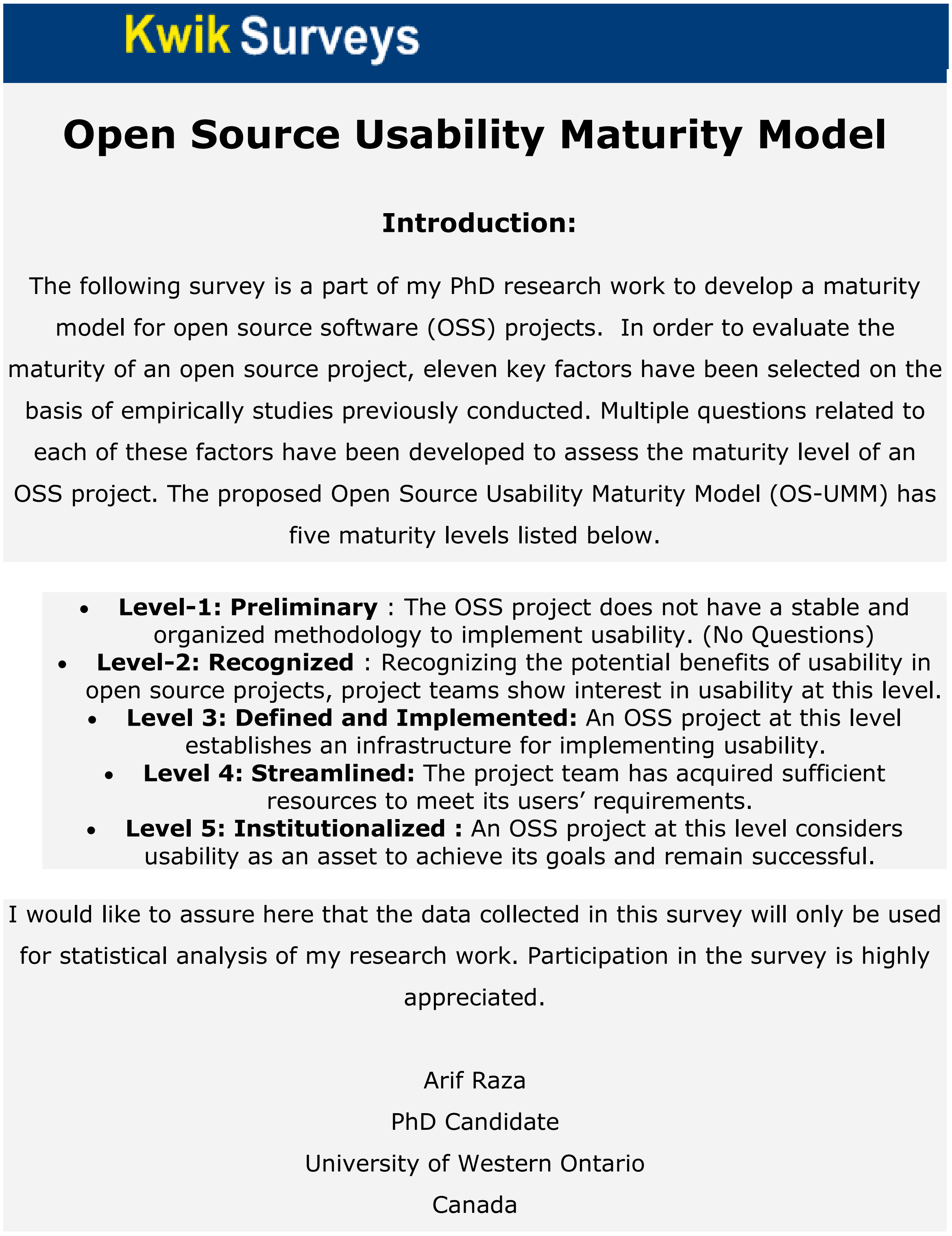
- **Understandability** ("*The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.*"[1])
- **Learnability** ("*The capability of the software product to enable the user to learn its application*."[1])
- **Operability (**"*The capability of the software product to enable the user to operate and control it*."[1])
- **Attractiveness (**"*The capability of the software product to be attractive to the user.*"[1])

I would like to assure here that the data collected in this survey will only be used for statistical analysis of my research work. Hence no information regarding any sort of identification is required.

Participation in the survey is highly appreciated.
Arif Raza (PhD Student)
University of Western Ontario
Canada

Appendix B

Screen Shot of OS-UMM Survey



Kwik Surveys

Open Source Usability Maturity Model

Introduction:

The following survey is a part of my PhD research work to develop a maturity model for open source software (OSS) projects. In order to evaluate the maturity of an open source project, eleven key factors have been selected on the basis of empirically studies previously conducted. Multiple questions related to each of these factors have been developed to assess the maturity level of an OSS project. The proposed Open Source Usability Maturity Model (OS-UMM) has five maturity levels listed below.

- **Level-1: Preliminary** : The OSS project does not have a stable and organized methodology to implement usability. (No Questions)
- **Level-2: Recognized** : Recognizing the potential benefits of usability in open source projects, project teams show interest in usability at this level.
- **Level 3: Defined and Implemented:** An OSS project at this level establishes an infrastructure for implementing usability.
- **Level 4: Streamlined:** The project team has acquired sufficient resources to meet its users' requirements.
- **Level 5: Institutionalized :** An OSS project at this level considers usability as an asset to achieve its goals and remain successful.

I would like to assure here that the data collected in this survey will only be used for statistical analysis of my research work. Participation in the survey is highly appreciated.

Arif Raza

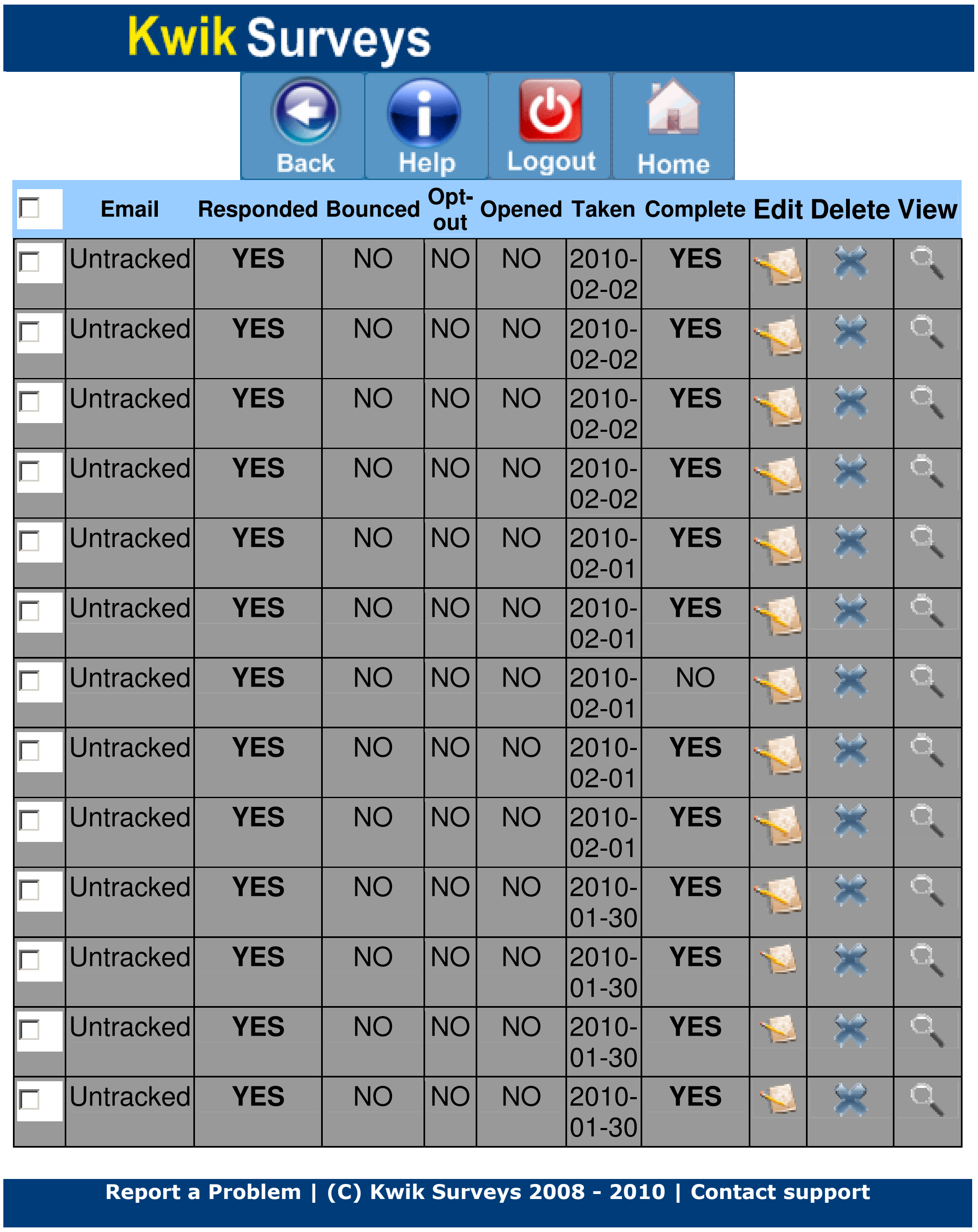
PhD Candidate

University of Western Ontario

Canada

Appendix C

Sample Survey Results in a Table



☐	Email	Responded	Bounced	Opt-out	Opened	Taken	Complete	Edit	Delete	View
☐	Untracked	**YES**	NO	NO	NO	2010-02-02	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-02	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-02	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-02	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-01	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-01	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-01	NO	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-01	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-02-01	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-01-30	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-01-30	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-01-30	**YES**	✎	✖	🔍
☐	Untracked	**YES**	NO	NO	NO	2010-01-30	**YES**	✎	✖	🔍

Report a Problem | (C) Kwik Surveys 2008 - 2010 | Contact support

Curriculum Vitae

Name:	Arif Raza
Post-secondary Education and Degrees:	University College of Engineering Taxila, Pakistan 1985 – 1990 B.Sc. Engineering (Civil)
	Birkbeck College University of London London, U.K. 1992-1994 M.Sc. (Computing Science)
Honours and Awards:	Silver Medalist B.Sc. Engineering (Honors) 1990
Related Work Experience	Lecturer KRL Model College Rawalpindi, Pakistan 1994-1998
	Assistant Professor National University of Sciences and Technology Rawalpindi, Pakistan 1998-2010, Study Leave
	Teaching Assistant The University of Western Ontario London, Ontario, Canada 2007-2010

Publications:

Raza, A. and Capretz, L.F. 2010. Contributors' preference in open source software usability: An empirical study, International Journal of Software Engineering & Applications (IJSEA), 1(2), 45 – 64.

Raza, A., Capretz, L.F. and Ahmed, F. 2010. Improvement of Open Source Software Usability: An Empirical Evaluation from Developers Perspective, Advances in Software Engineering, vol. 2010, Article ID 517532, 12 pages, 2010. doi:10.1155/2010/517532

Raza, A., Capretz, L.F. and Ahmed, F. 2010. Users' perception of open source usability:An empirical study, Engineering with Computers, Under Review

Raza, A. and Capretz, L.F. 2010. An empirical study of open source software usability - The industrial perspective, International Journal of Open Source Software and Processes, Under Review

Raza, A., Capretz, L.F. and Ahmed, F. 2010. An open source usability maturity model (OS-UMM), Behaviour & Information Technology, Under Review

Raza, A. and Capretz, L.F. 2009.Usability as a dominant quality attribute, Proceedings of the 2009 International Conference on Software Engineering Research Practice, SERP 2009, July 13-16, 2009, Las Vegas Nevada, USA, 571-575

Ahmed, F. and Raza, A. 2003. Survivability as an integral part of software testing strategy, Proceedings of the International Conference on Computer Science, Software Engineering, Information Technology, e-Business, and Applications, Rio de Janeiro, 121-125