Computers in Human Behavior 28 (2012) 1109–1121

Contents lists available at SciVerse ScienceDirect

Computers in Human Behavior

journal homepage: www.elsevier.com/locate/comphumbeh

An open source usability maturity model (OS-UMM)

Arif Raza [a,*], Luiz Fernando Capretz [a], Faheem Ahmed [b]

[a] Department of Electrical & Computer Engineering, University of Western Ontario, London, Ontario, Canada N6A 5B9
[b] Faculty of Information Technology, United Arab Emirates University, P. O. Box 17551, Al Ain, United Arab Emirates

ARTICLE INFO

Article history:
Available online 12 February 2012

Keywords:
Open source software
Usability evaluation
Maturity model
Empirical analysis

ABSTRACT

User satisfaction has always been a major factor in the success of software, regardless of whether it is closed proprietary or open source software (OSS). 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 paper presents a usability maturity model specifically aimed at usability-related issues for open source projects. In particular, the model examines the degree of coordination 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 paper discusses assessment questionnaires, a rating methodology and two case studies.

1. Introduction

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) 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. With the involvement of and acceptance from large commercial IT vendors, OSS products have transitioned from a fringe activity into the mainstream software culture (Fitzgerald, 2006). Researchers who study OSS quality -related issues such as quality control, quality assurance techniques, and risk assessment, testing and usability agree that OSS quality-related issues differ significantly from those of closed proprietary software (Benson, Muller-Prove, & Mzourek, 2004; Çetin & Göktürk, 2007; Crowston, Annabi, & Howison, 2003).

In the ISO 9241-11 (1998) 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*." The International Organization for Standardization and The International Electro technical Commission ISO/IEC 9126-1 (2001) categorizes software quality attributes into six categories: functionality, reliability, usability, efficiency, maintainability and portability. Alternatively, this particular standard defines usability as "*the capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions*". However, Bevan (2009) maintains 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*." He also identifies four human-centered activities for software design, which include understanding and specifying the context of use, specifying the user requirements, design solutions and evaluation.

According to Gill (1996), user satisfaction with a system can be enhanced through intrinsic motivational factors such as increased sense of user control, more task variety, less task routine, and providing capabilities to move task performance to higher levels. Oviatt and Cohen (2000) report that a profound shift is now occurring toward embracing users' natural behavior as the center of the human–computer interface. Picard (1997) suggested several applications where it is beneficial for computers to recognize human emotions for example, knowing the user's emotions; the computer can become a more effective tutor. According to Bianchi-Berthouze and Lisetti (2002) three key points are very important when developing systems that capture affective information: embodiment (experiencing physical reality), dynamics (mapping the experience and the emotional state onto a temporal process and a particular label), and adaptive interaction (conveying emotive response, responding to a recognized emotional state).

Lieberman, Paternò, and Wulf (2006) emphasize user participation in earlier stage of software design. However they realize that specific categorization of user requirements would not be easy due to their diversity and on-going changes. They thus stress end

* Corresponding author. Address: Department of Electrical & Computer Engineering, University of Western Ontario, London, Ontario, Canada N6A 5B9. Tel.: +1 226 688 5372; fax: +1 519 8502436.
E-mail addresses: araza7@uwo.ca (A. Raza), lcapretz@uwo.ca (L.F. Capretz), f.ahmed@uaeu.ac.ae (F. Ahmed).

0747-5632/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.chb.2012.01.018

users to show flexibility in adapting the system. The authors desire that users should adjust themselves to adapt systems according to their skill level. Underscoring human emotions in human computer interaction, Cristescu (2008) highlight the issues of "*emotional design approach, the importance of nonverbal as an instrument of usability evaluation and the role of emotions in human–computer interaction.*" The author observes that users might get frustrated or annoyed with any new complex interface. The author proposes the nonverbal as a usability evaluation tool to avoid such problems. Barbeite and Weiss (2004) consider "anxiety" and "self-efficacy" as the two main parameters to determine human behavior. In the study, self-efficacy is considered an important predictor of computer use whereas anxiety is considered a predictor of self-efficacy and is correlated with different indicators of computer usage.

Iivari, Hedberg, and Kirves (2008) observe that OSS is targeted to the general public. Highlighting the importance of usability in businesses, they observe that *"typically, the OSS developers do not have knowledge about the non technical users, their tasks and the context of use."* Hence, these authors suggest the necessity of expert opinions as well as realistic user feedback at an earlier design stage. Çetin and Göktürk (2007) believe that traditionally, usability and user-centric designs have been challenges in open source software. Nevertheless, they indicate their optimistic belief that usability is beginning to become institutionalized in open source projects. On the other hand, Zhao and Deek (2005) identify a weak relationship between OSS and usability. Specifically, they observe that requests of experienced users of OSS do not necessarily reflect the requirements of non-experienced users. Furthermore, they highlight the lack of resources to conduct formal usability and emphasize the importance of building effective communication systems to involve novice users and to provide them with usability knowledge. Andreasen, Nielsen, Schrøder, and Stage (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. Moreover, Hedberg, Iivari, Rajanen, and Harjumaa (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 a non-technical user.

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. The measuring instrument of the model contains factors that have been selected from four of our empirical studies, which examine the perspectives of OSS contributors, users, developers, and the industry (Raza & Capretz, 2010; Raza, Capretz, & Ahmed, 2010a, 2010b, 2011). To the best of our knowledge, this is the first study of its kind within the open source field. The model has been developed in response to a need for measuring how well open source software projects support usability. Specifically, it is intended for use in assessing and improving the usability aspect in open source software development.

In the next section, we present a literature review that is related to the usability testing, usability models and maturity models that motivated this research. Section 3 illustrates the framework of the open source usability maturity model (OSUMM). Subsequently, Section 4 explains the performance scale and the rating methodology for the model. In Section 5, the reliability and validity analyses of questionnaires is presented. Section 6 discusses the case studies and the assessment methodology of the model. In Section 7, we discuss the achievements as well as limitations of the model and of this research work in general. Finally, Section 8 provides a conclusion to the paper.

2. Literature review

2.1. General usability assessment and measurement

Aspects of usability cannot be improved unless there are ways to test and measure them. While studying current practices in usability measurement, Hornbæk (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 a journey 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, broad 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 the testing of software 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.

Aberdour (2007) contrasts the *"formal and structured testing"* that is typical in closed software development with the *"unstructured and informal testing"* in OSS development. Finally, Hedberg et al. (2007) identify that the processes of *"test coverage, test-driven development and testing performed by developers"* require more attention in OSS projects through formal and sufficient test plans that ensure errors are caught before the release of the software.

2.2. Usability issues in open source projects

The facts that the users of OSS are no longer limited to developers and the software quality is determined by end user's experience, make the usability an important quality attribute than it is generally realized. In an empirical study related to user participation, Iivari (2009) observes that users have informative, consultative and participative roles in OSS projects. Also, Bevan (2008) recommends the incorporation of human-centered design resources in the earlier stages of the software life cycle.

In an empirical study related to usability issues in OSS, Andreasen, Nielsen, Schrøder, and Stage (2006) realize that both the user-centered design strategy and the usability engineering has largely been neglected in OSS research and development. While they find that OSS developers are interested in usability, it is neither considered one of the top priorities nor adopted by any systemic evaluation method. Although these authors recognize the increasing awareness among OSS developers, they identify a gap between the technical contributors of OSS and the usability professionals, and hence, they emphasize the need for a greater focus on usability in open source projects.

In examining the usability errors reported by OSS users, Zhao and Deek (2006) observe that the average user does not know

how to effectively report these errors. Nichols and Twidale (2006) identify similar difficulties faced by users in reporting usability bugs. Specifically, they observe that usability issues, as expected, are more subjective in nature and could prolong the process of analyzing and fixing the related bugs.

Moreover, Raza and Capretz (2009) maintain that software developers have an egocentric viewpoint of their designs; the developers consider themselves as the end users and believe that a design they consider acceptable will be sufficient for the targeted users.

Abran et al. (2003) do not consider software usability as a luxury; rather, they consider it as a prime attribute of productivity and software acceptance. According to these authors, usability has different meanings for each stakeholder, as it is a "*determinant of performance*" for end users, *"a major decision point in selecting a product"* for managers, and it entails *"issues like design quality, documentation maintainability"* for software developers.

According to Zhao and Deek (2006), "*developing an effective method for learning usability inspection is necessary for non-expert inspectors to contribute to OSS usability improvement*." In particular, these authors consider the exploratory learning method as one of the best methods, as it allows people to learn according to their own understanding. In conclusion, they state the existence of a positive relationship between their proposed exploratory learning methods and the effectiveness of usability inspection; nevertheless, the conclusion is limited by the scope of their study.

2.3. Usability models

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

Winter, Wagner, and Deissenboeck (2008) propose a usability model relating system properties to user activities. The authors claim that their model *"fosters preciseness and completeness,"* provides a basis for analysis and measurement and describes the system's usability in relation to the activities performed by the user. However, as the authors admit, the model requires refinement to more specific contexts.

Çetin and Göktürk (2008) stress the importance of testing and measurement by stating that "*one can't improve what is not measured*." Accordingly, the authors propose 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.

Moraga, Calero, Piattini, and Diaz (2007) propose a usability model for portlets and utilize different attributes from ISO-9126 that affect usability. In its application to real portlet, the adequacies of the portlet, along with some weaknesses related to usability, have been identified. Currently, the authors plan to complete a validation of the presented model.

Seffah, Donyaee, Kline, and Padda (2006) have developed the Quality in Use Integrated Measurement (QUIM) model for measuring software usability. This hierarchical model decomposes usability into a series of sub-factors that are quantifiable by means of one or more specific metrics, which consist of either a formula or countable data. However, considering their model as a first step towards their goal, the authors state *"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."*

In their effort to present a consolidated usability model, Abran et al. (2003) present an analysis of ISO-9241-11 and ISO-9126. Consequently, they discover that ISO-9241-11 was developed by human computer interaction (HCI) specialists, whereas ISO-9126 resulted from the efforts of software engineering (SE) experts. The authors propose a revised integrated usability model that includes *"learnability"* and *"security"* as their baseline from ISO-9241 and the structured hierarchy from ISO-9126. However, they realize more work needs to be done in the area of usability modeling; specifically, it needs more consensus among researchers and requires a more comprehensive model.

Granollers, Lorés, and Perdrix (2003) have created The Usability Engineering Process Model (UEPM), which integrates SE with HCI activities and focuses upon usability. Specifically, they discuss 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."* The authors consider the validation of the model by real world projects as vital, and accordingly, they state that they are in the process of developing different paradigm applications.

2.4. Maturity models

Software process assessment models, such as CMM (Paulk, Weber, Curtis, & Chrissis, 1995) and CMMI (Chrissis, Konran, & Shrum, 2006) define maturity levels in stages with the objective of qualitatively illustrating the maturity of the software engineering process.

In his report, Earthy (1999) presents a "*Usability Maturity Model: Processes*" to assess an organization's capability of performing activities related to human-centered processes. Specifically, the model consists of seven possible processes that may occur during system development. Furthermore, conforming to ISO 15504, this model describes six capability levels, as listed in Table 1.

Earthy's *"Usability Maturity Model: Human Centredness Scale"* (1998) presents how organizations could progress through six levels of human-centered processes. Additionally, a rating methodology assesses *"the level of maturity reached by an organisation in its capability to do human-centred design."* The maturity scale accords to ISO 15504, SPICE.

Lethbridge (2007) presents the User and Usability Maturity Model (UUMM), which has been modeled after the Capability Maturity Model, and which can assess an organization's capabilities with users and usability-related issues. The model covers four dimensions, including *"involvement with users, training of the team, development processes and evaluation processes."* However, the authors have not yet used the model to conduct an assessment of an organization.

Table 1 summarizes the existing maturity models and their respective scales. As previously discussed, none of these scales describe the maturity of software processes in terms of issues specifically related to open source projects. As a result, our work expands upon the concepts presented in these studies by proposing a scale that reflects the usability maturity of an open source software project; the proposed maturity scale includes key usability factors, such as User Requirements, User Feedback, Usability Learning, User-Centered Design Methodologies, Understandability, Learnability, Operability, Attractiveness, Usability Bug Reporting, Usability Testing and Documentation. Our maturity scale includes five levels, which, in ascending order, include *Preliminary, Recognized, Defined, Streamlined and Institutionalized.* Accordingly, this maturity scale results in a framework, as described below, which consists of a set of questionnaires for each level. The set of statements in the questionnaires are further subdivided into eleven usability factors and incorporate four usability dimensions.

3. Framework of OS-UMM

In comparison to other software issues, usability issues are more subjective in nature and hence more debatable. For example,

Table 1
Maturity models and their scales.

CMM (Paulk et al., 1995)	CMMI (Chrissis et al., 2006)	UUMM (Lethbridge et al., 2007)	Usability maturity model: Processes (Earthy, 1999)	Usability maturity model: human centredness scale (Earthy, 1998)
Initial	Initial	Haphazard	Incomplete	Unrecognized
Repeatable	Managed	Defined input from users and usability awareness	Performed	Recognized
Defined	Defined	Iterative interactions with users and design for usability	Managed	Considered
Managed	Quantitatively Managed	Controlled and measured involvement of users	Established	Implemented
Optimizing	Optimizing	Continually improving usability	Predictable Optimizing	Integrated Institutionalized

a user interface (UI) element may be confusing to some people and clear to others. However, usability aspects nevertheless need to be tested and measured. The increasing popularity of usability assessment in open source software projects necessitates a usability maturity evaluation methodology. To the best of our knowledge, no usability maturity model has been created for OSS. Consequently, we present a usability assessment methodology for OSS projects, which is the first study of its kind in the area of open source software.

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 immensely important topic of usability assessment. As already mentioned above, in four of our empirical studies from OSS users, developers, contributors and industry members, we have been able to identify eighteen 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 some overlapping factors, such as User Requirements, User Expectations, Incremental Design Approach, Knowledge of UCD Methods, Usability Testing and Usability Assessment, we have obtained eleven key usability practices, which are depicted in Table 2.

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 life cycle 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 2
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.	User-Centered Design (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 2, 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 User Requirements, User 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: User Requirements (URs), User Feedback (UFB), Usability Learning (UL), User-Centered Design Methodology (UCD), Understandability (U), Learnability (L), Operability (O), Attractiveness (A), 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).
SN = Statement Number (an integer).

3.1. 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-centered 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.

3.2. 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 show 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 in Appendix A.

3.3. 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 fulfil its users' requirements and expectations. Furthermore, they are able to collect feedback from users, by utilizing a planned strategy to improve 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. Essential technical skills are also adopted by the project team 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 in Appendix B.

3.4. 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 the 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. Appendix C illustrates the measuring instrument for the usability maturity of an OSS project when it is at the *"Streamlined"* level.

3.5. 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 software 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 in Appendix D.

Table 3 summarizes the number of items in assessment questionnaires for each usability factor related to all the five maturity levels.

4. Performance scale and rating methodology

4.1. Performance scale

The maturity level of an OSS project is determined by its ability to demonstrate the usability factors. In order to determine this level, we have used a five-level scale to rate a project's performance. The rating is a quantitative indication of the extent to which a project agrees with each statement in the questionnaires, and, more specifically, the way in which a project fulfils the requirements of a specific maturity level. Table 4 shows the ordinal ratings used to measure each usability factor, including *"Fulfilled"*, *"Largely Fulfilled"*, *"Partially Fulfilled"*, *"Not Fulfilled"* and *"Not Applicable."* To increase the flexibility of our methodology, the rating of *"Not Applicable"* has been included in the model. In order to maintain the consistency of our usability assessment with the already accepted and validated popular scales, we have structured the performance scales and their thresholds close to the BOOTSTRAP methodology (Wang & King, 2000). However, the linguistic expressions have been slightly modified according to the design of the questionnaires in our model, OS-UMM. 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, our methodology uses the self-assessment approach.

4.2. Rating methodology

As previously mentioned, we have partially derived the rating methodology from the BOOTSTRAP algorithm (Wang & King, 2000). We have used terms, such as Usability Rating (UR_{UM}), Number of Fulfilled Statements (NF_{UM}), Passing Threshold (PT_{UM}) and Usability Maturity Level (UML).

Let UR_{UM} $[i, j]$ be a rating of the ith usability factor of the jth maturity level. Subsequently, according to the scales defined in Table 4, it can be summarized as:

Table 3
Framework of OS-UMM.

Maturity level	Usability factors and number of items in assessment questionnaire											
	UR	UF	UL	UCD	U	L	O	A	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

Table 4
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	–	–

$UR_{UM}[i, j]$	=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 $UR_{UM}[i, j] \geqslant 3$ or $UR_{UM}[i, j]$ is 0. The number of conditions/statements fulfilled at jth maturity level is defined as:

$NF_{UM}[j]$	=Number of $\{UR_{UM}[i, j] \mid \text{Fulfilled}\}$ =Number of $\{UR_{UM}[i, j] \mid UR_{UM}[i, j] \geqslant 3$ or $UR_{UM}[i, j] = 0\}$

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

$$PT_{UM}[j] = N_{UM}[j] * 80\%$$

With the values calculated to the nearest ten, the passing threshold of 80% at each Usability Maturity Level is illustrated in Table 5.

Table 5
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 conditions or statements fulfilled is greater than or equal to the passing threshold PTUM[j]; hence:

$$UML = \max\{j \mid NF_{UM}[j] \geqslant PT_{UM}[j]\}$$

5. Reliability and validity analysis of questionnaires

The two integral features of any empirical study are reliability and validity. Reliability refers to the consistency of the measurement, and validity is the extent to which a measurement reflects the true value. In order to conduct a pilot study, we selected active projects from sourceforge.net that had an activity level of 90% or greater, in different categories. First of all, we established contacts with project managers in these open source projects. In particular, we sent 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 designed. For this empirical investigation, we used the most common approaches in for conducting reliability and validity analysis of the measuring instruments. The reliability of the multiple-item measurement scales for the five maturity levels is 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 6. Nunnally and Bernste (1994) maintain that a reliability coefficient of 0.70 or higher for a measuring instrument 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 some of the items with an alpha coefficient of less than 0.70 still fall within the acceptable range of van de Ven and Ferry (1980) and Osterhof (2001). Therefore, based on the criteria for reliability, we conclude that all the items developed for this empirical investigation are reliable.

Table 6
Reliability analysis of usability factors.

Maturity level	Usability factors										
	UR	UF	UL	UCD	U	L	O	A	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

Table 7
Construct validity of usability factors.

Maturity level	Usability factors										
	UR	UF	UL	UCD	U	L	O	A	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

Campbell and Fiske (1959) state that convergent validity occurs when scale items correlate and move in the same direction for a given assembly. Principal component analysis (Comrey & Lee, 1992) is performed for all eleven usability factors in each maturity level and reported in Table 7. Specifically, we have utilized the Eigen value (Kaiser, 1970) as a reference point to observe 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 questionnaires completely form a single factor. Therefore, we conclude that the convergent validity can be regarded as sufficient.

6. Case studies

6.1. Assessment methodology

In order to perform the project's usability maturity assessment, we applied our model to two OSS projects. To protect the privacy of the two projects, they will be referred to as "A" and "B." These OSS projects are conscious of and realize the importance of usability as mentioned on their websites.

The participants were informed that the assessment was a part of a research study and neither the identity of an individual nor of a project would be disclosed 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 a Likert scale ranging from 0 to 4, as illustrated in Table 4. 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. Keeping up the spirit of OSS projects, all the 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 the case studies are discussed in the following sections. 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 following section.

6.2. Case Study – Project "A"

Project "A" falls under the category of Desktop Environment and deals with multiple related issues. The project has in excess of 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 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 8. According to the rating method discussed in Section 4.2, a statement is considered agreed upon if the performance scale shown in Table 4 is either greater than or equal to 3 or 0. From the data presented in Table 8, we have thus computed NF_{UM} (the Number of Fulfilled Statements), for each level. For Level 2, NF_{UM} is 23, for Level 3, it is 9, for Level 4 it is 6 and for Level 5, NF_{UM} is 5. According to the rating threshold for our OS-UMM, NF_{UM} at Level 2 has a pass threshold of 80%. Consequently, Project A is therefore at the *"Recognized"* maturity level, or Level 2.

6.3. 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 another with help and support discussion forum.

The numerical values entered into each cell of Table 9 represent the extent to which Project B corresponds to the statements in the questionnaires of each maturity level. Accordingly, we compute that for Level 2, NF_{UM} is 24, for Level 3, it is 19, for Level 4 it is 18 and for Level 5, NF_{UM} is 17. According to the rating threshold for our OS-UMM, NF_{UM} at Level 2 has a pass threshold of 80%. Based on the rating method discussed in Section 4.2, Project-B is also at Level 2, or, the *"Recognized"* level. Table 10 summarizes the assessment results for both case studies.

6.4. 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 the same processes. In order to evaluate inter-rater agreement, The Kendall coefficient of concordance (von Eye & Mun, 2005) is often used. However, Cohen's Kappa (Cohen, 1960) is preferred to 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 Project A and Project B were 4 and 6 respectively.

Table 11 reports the Kendall and Kappa statistics for Project A. The values of Kendall's coefficient, Cohen's Kappa and the Fleiss Kappa coefficients can range from 0 to 1, with 0 indicating perfect disagreement and 1 indicating perfect agreement (Landis & Koch, 1977). For Kendall's coefficient, higher values indicate a stronger association. In Project A, the Kendall's coefficients range from

Table 8
Details of assessment results of Case Study A.

Recognized Level-2		Defined Level-3		Streamlined Level-4		Institutionalized 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				

Table 9
Details of assessment results of Case Study B.

Recognized Level-2		Defined Level-3		Streamlined Level-4		Institutionalized 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				

0.95 to 0.99, as shown in Table 11. The benchmark for Kappa (El Emam, 1999) includes four level scales, 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,

Table 10
Summary of assessment results of case studies.

Usability maturity level	Total questions	Pass threshold 80%	Project-A NF_{UM}	Project-B NF_{UM}
Preliminary	None	–	–	–
Recognized	29	23	23	24
Defined	29	23	9	19
Streamlined	28	22	6	18
Institutionalized	25	20	5	17

Table 11
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*

* $P < 0.001$.
** $P < 0.005$.

Table 12
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*

* $P < 0.001$.
** $P < 0.005$.

and hence, fall into the category of substantial. For Project B, the Kendall's coefficients range from 0.95 to 0.99, and the Kappa coefficients range from 0.62 to 0.68, as shown in Table 12. Hence, as in the case of Project A, these coefficients are categorized as being substantial.

7. 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.

The experiences of end users have profound impacts on the success of a software project, and, as a result, usability and its related issues are a key area of research in the open source community. In order to identify precise areas where improvement is necessary, assessment needs to be performed. OSS is a relatively new software area, having a history of producing software "*by engineers for engineers*" (Benson et al., 2004). However, 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.

Our previous work utilized empirical investigation to examine the impact of some key factors of OSS usability improvement. Additionally, research models have been developed to study and establish the relationship between key usability factors from the perspective of different stakeholders. For the usability assessment methodology of open source projects, the significant key factors have been used as a measuring instrument to present a *Usability Maturity Model for OSS projects:* OS-UMM. The structural composition of OS-UMM consists of assessment frameworks from four dimensions based on the perspectives of OSS developers, users and contributors, who include architects, designers, developers, testers and users, as well as the software industry.

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.

7.1. 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.

8. Conclusion

Our 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 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. The framework of the model is comprised of assessment questionnaires for four of the five maturity levels, as well as a performance scale and a rating methodology. Additionally, two case studies have been discussed, as we have examined the performance of two OSS projects in the area of usability. Apart from its limitations, this work contributes towards establishing a comprehensive strategy for the usability maturity of OSS projects and addresses the important issue of usability assessment in open source projects.

Appendix A. The measuring instrument for assessing the usability maturity of an OSS project, when it is at Level-2

UF.2.1 *Users' Requirements*
S.2.1.1 Project designers and developers agree that the only way to enhance software acceptance level is to address users' requirements
S.2.1.2 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 handled and learned by sharing each other's viewpoints and open discussion
S.2.3.3 Plans are developed to have formal training for at least 20% of the developers, in learning usability design principles

UF.2.4 *User Centered Design (UCD) Methodology*
S.2.4.1 Project team recognizes the importance of 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% members of the software development team members have back ground knowledge about UCD methods

UF.2.5 *Understandability*
S.2.5.1 Project designers and developers are acquiring knowledge on how to increase understandability of their product among the users
S.2.5.2 OSS team still lacks systematic strategy to enhance understandability level of their software project considering their target users limitations

UF.2.6 *Learnability*
S.2.6.1 Project team realizes the need of increasing learnability of their product for their prospective users
S.2.6.2 There is a lack of technical knowledge among team members about how to increase learnability of their software
S.2.6.3 Project team considers increased learnability as an option to reduce the number of complaints from users

UF.2.7 *Operability*
S.2.7.1 Project developers agree in principle, that what is operable for them may not be operable for end users
S.2.7.2 Understanding the importance, project team realizes the need of increasing operability of their product
S.2.7.3 Developers are working on a strategic plan to enhance operability of their software project

UF.2.8 *Attractiveness*
S.2.8.1 Project team collects information on how to enhance attractiveness of their product
S.2.8.2 There is still a lack of defined strategic plan to increase the project's attractiveness
S.2.8.3 Project team promotes innovative ideas to make their product pleasing for its users

UF.2.9 *Usability Bug Reporting*
S.2.9.1 Difficulties faced by users in reporting usability related errors is realized by the development team
S.2.9.2 Project development team is working on plans to provide end users with a convenient way to report bugs and errors

UF.2.10 *Usability Testing*
S.2.10.1 The project team realizes the need of 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 Project team collects information on how to ensure comprehensive assessment of functional and non functional requirements

UF.2.11 *Documentation*
S.2.11.1 Project team understands and realizes the importance of documentation, however still lacks systematic and planned strategy to do that
S.2.11.2 The project has no formal documentation of users' requirements, design, testing and future improvement plans

Appendix B

B.1. Level 3: Defined

UF.3.1 *Users' Requirements*
S.3.1.1 Identifying target users and collecting their requirements are considered essential to achieve 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 Project team is committed to acquire knowledge about usability engineering
S.3.3.2 Project has adequate resources to allocate for usability learning
S.3.3.3 At least 20% of the project developers have formal training in learning usability design principles

UF.3.4 *User Centered Design (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 back ground knowledge of UCD methods

UF.3.5 *Understandability*
S.3.5.1 OSS team has defined a systematic strategy to enhance understandability level of their software project
S.3.5.2 Metrics have been defined to measure understandability level of the project

UF.3.6 *Learnability*
S.3.6.1 Project team has adopted a well designed methodology to enhance learnability of their software
S.3.6.2 At least 50% of team developers have sufficient knowledge and technical abilities to improve learnability of the project

UF.3.7 *Operability*
S.3.7.1 Weak areas related to project's operability are identified and necessary steps are taken for improvement
S.3.7.2 Keeping in mind user limitations, the project team has developed a strategic plan to enhance operability of their software project

UF.3.8 *Attractiveness*
S.3.8.1 A strategic plan is defined to increase attractiveness of the project
S.3.8.2 Project team makes use of innovative ideas to make 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 technical abilities to provide convenient usability bug reporting facility to their users
S.3.9.2 Project team is committed to improve usability bug reporting facility to their users
S.3.9.3 Difficulties faced by users in reporting usability related errors is recorded and maintained by the development team

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 to assess 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 Project team has formal documentation of user requirements, design and testing
S.3.11.2 Resources have been allocated for project documentation and its maintenance
S.3.11.3 The project has well documented future improvement plans

Appendix C

C.1. Level 4: Streamlined

UF.4.1 *Users' Requirements*
S.4.1.1 The project has adequate resources and skills to gather user requirements and is able to come up to 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 Project team learns from user feedback and avoids repeating previous mistakes
S.4.2.2 A well established management system of the project records user feedback regularly and takes appropriate actions on its basis

UF.4.3 *Usability Learning*
S.4.3.1 Formal and informal methods are used by project developers 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 *User Centered Design (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 innovative use of UCD methods
S.4.4.3 The development team members consult HCI and usability experts regularly regarding UCD principles and their implementation

UF.4.5 *Understandability*
S.4.5.1 The understandability among the users is regularly monitored through user feedback
S.4.5.2 Through already defined metrics, understandability level of the project is measured
S.4.5.3 Necessary steps are taken, recorded and monitored to enhance 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 operability of the software project
S.4.7.2 Modularized system design is followed in the software project to make it more operable
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 attractiveness of the project

(*continued on next page*)

S.4.8.2 Project team regularly monitors the outcome of innovative ideas to make their product pleasing
S.4.8.3 User feedback is collected and maintained regularly about the software's attractiveness

UF.4.9 *Usability Bug Reporting*
S.4.9.1 Project development team has developed a service protocol to provide end users with a convenient way to report usability bugs and errors
S.4.9.2 Project team monitors the use of the developed service protocol for effective and easy way of reporting usability bugs

UF.4.10 *Usability Testing*
S.4.10.1 The project testers learn from the previous experience and test results and 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 do usability testing regularly

UF.4.11 *Documentation*
S.4.11.1 Project documentation has a clear set of guidelines to handle usability related issues
S.4.11.2 Project documentation provides a checklist to inspect usability related issues

Appendix D

D.1. Level 5: Institutionalized

UF.5.1 *Users' Requirements*
S.5.1.1 The project successfully responds to the user requirements and come up to the expectations of its users
S.5.1.2 Incorporating user requirements has become essential part of the project 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 software project's quality in general and usability in particular
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 improving knowledge in the area of HCI and usability
S.5.3.2 The project developers successfully employ innovative ideas based on their usability knowledge

UF.5.4 *User Centered Design (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 User feedback indicates their satisfaction and ability to conveniently understand the software features
S.5.5.2 The defined strategy to enhance understandability level of the project is regularly reviewed and updated as required

UF.5.6 *Learnability*
S.5.6.1 Interactive help is provided to end users to enhance their learnability of the software project
S.5.6.2 Metrics are used to measure learnability level regarding any new features in the software

UF.5.7 *Operability*
S.5.7.1 Advance features are introduced in software in a gradual way to make it more operable and give the users more control
S.5.7.2 Defined metrics are used to identify weaknesses, quantitatively measure and 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 user interface pleasing and attractive for the users
S.5.8.2 Regular monitoring of the strategic plan, its outcome related to the project's attractiveness and its improvement are the parts of continuous strategic efforts of the team

UF.5.9 *Usability Bug Reporting*
S.5.9.1 Project team regularly responds, maintains and improves the already developed service protocol for 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 through innovative methods to continuously improve the process of usability testing
S.5.10.2 Quantifiable metrics are used to measure usability effectively and objectively
S.5.10.3 A usability test management system keeps track of usability tests and uses the results to improve software quality and usability of the project

UF.5.11 *Documentation*
S.5.11.1 Project documentation is updated on regular basis to reflect any modification
S.5.11.2 A log has been maintained to record user complaints regarding usability related issues and the actions taken upon them

References

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

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

Andreasen, M. S., Nielsen, H. V., Schrøder, S. O., & Stage, J. (2007). What happened to remote usability testing? An empirical study of three methods. The SIGCHI conference on human factors in computing systems.

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

Barbeite, F. G., & Weiss, E. M. (2004). Computer self-efficacy and anxiety scales for an Internet sample: Testing measurement equivalence of existing measures and development of new scales. *Computers in Human Behaviour, 20*(1), 1–15.
Benson, C., Muller-Prove, M., & 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, 2004.
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.
Bianchi-Berthouze, N., & Lisetti, C. (2002). Modeling multimodal expression of user's affective subjective experience. *User Modeling and User adapted. Interaction, 12*, 49–84.
Campbell, D. T., & Fiske, D. W. (1959). Convergent and discriminant validation by the multi-trait multi-method matrix. *Psychological Bulletin, 56*(2), 81–105.
Çetin, G., & 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, 2008.
Çetin, G., & Göktürk, M. (2007). Usability in open source community. *ACM Interactions, 14*(6), 38–40.
Chrissis, M. B., Konran, M., & Shrum, S. (2006). *CMMI: Guidelines for process integration and product improvement* (2nd ed.). Addison–Wesley Publishing Company, pp. 52–69.
Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*, 37–46.
Comrey, A. L., & Lee, H. B. (1992). *A first course on factor analysis* (2nd ed.). Hillsdale.
Craven, J., & Booth, H. (2006). Putting awareness into practice: Practical steps for conducting usability tests. *Library Review, 55*(3), 179–194.
Cristescu, I. (2008). Emotions in human-computer interaction: The role of nonverbal behavior in interactive systems. *Revista Informatica Economică, 2*(46), 110–116.
Cronbach, L. J. (1951). Coefficient alpha and the internal consistency of tests. *Psychometrica, 16*, 297–334.
Crowston, K., Annabi, H., Howison, J. (2003). Defining open source software project success. 24th International conference on information systems (ICIS), Seattle, WA.
Earthy, J. (1998). Usability maturity model: Human-centredness scale. IE2016 INUSE Deliverable D5.1.4s, 1998.
Earthy, J., (1999). Usability maturity model: Processes: Version 2.2. Lloyd's Register, London.
El Emam, K. (1999). Benchmarking kappa: Inter-rater agreement in software process assessments. *Empirical Software Engineering, 4*(2), 113–133.
Fitzgerald, B. (2006). The transformation of open source software. *MIS Quarterly, 30*(3), 587–598.
Gill, T. G. (1996). Expert systems usage: Task change and intrinsic motivation (pp. 301–329). MIS Quarterly, Summer.
Granollers, T., Lorés, J., & Perdrix, F. (2003). Usability engineering process model. Integration with software engineering, HCI-Intl'03, Crete-Greece.
Hedberg, H., Iivari, N., Rajanen, M., & Harjumaa, L. (2007). Assuring quality and usability in open source software development. In The 1st international workshop on emerging trends in FLOSS research and development, FLOSS, IEEE Computer Society, Washington, DC, 2007, May 20–26.
Hornbæk, K. (2006). Current practice in measuring usability: Challenges to usability studies and research. *International Journal of Human–Computer Studies, 64*, 79–102.
Iivari, N. (2009). Constructing the users in open source software development: An interpretive case study of user participation. *Information Technology & People, 22*(2), 132–156.
Iivari, N., Hedberg, H., & 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.
ISO 9241 (1998). Ergonomics requirements for office with visual display terminals (VDTs).
International Standard ISO/IEC 9126-1 (2001). Software engineering – Product quality – Part 1: Quality model (1st ed., 2001-06-15), 9–10.
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.
Landis, J., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *Biometrics, 33*, 159–174.
Lee, H. Y., Jung, H. W., Chung, C. S., Lee, J. M., Lee, K. W., & Jeong, H. J. (2001). Analysis of inter-rater agreement in ISO/IEC 15504-based software process assessment. 2nd Asia–Pacific conference on quality software 2001, 341–348.
Lethbridge, T. (2007). UUMM: User and usability maturity model, personal notes. School of Information Technology and Engineering, University of Ottawa, 2007, Personal Information.
Lieberman, H., Paternò, F., & Wulf, V. (2006). *End user development: Empowering people to flexibly employ advanced information and communication technology*. Dordrecht: Springer.
Miller, J. (2006). Usability testing: A journey, not a destination. *Internet Computing, IEEE, 10*(6), 80–83.
Moraga, M. Á., Calero, C., Piattini, M., & Diaz, O. (2007). Improving a portlet usability model. *Software Quality Control, 15*(2), 155–177.
Nichols, D. M., & Twidale, M. B. (2006). Usability processes in open source projects. *Software Process: Improvement and Practice, 11*(2), 149–162.
Nunnally, J. C., & Bernste, I. A. (1994). *Psychometric theory* (3rd ed.). New York: McGraw Hill.
Osterhof, A. (2001). *Classroom applications of educational measurement*. NJ: Prentice Hall.
Oviatt, S. L., & Cohen, P. R. (2000). Multimodal systems that process what comes naturally. *Communications of the ACM, 43*(3), 45–53.
Paulk, M. C., Weber, C. V., Curtis, B., & Chrissis, M. B. (1995). *The capability maturity model: Guidelines for improving the software process/CMU/SEI*. Addison-Wesley Publishing Company, pp. 15–28.
Picard, R. W. (1997). *Affective computing*. MIT Press.
Raymond, E. S. (1999). *The cathedral and the bazaar*. Sebastopol, CA: O'Reilly.
Raza, A., & Capretz, L. F. (2009). Usability as a dominant quality attribute. *International Conference on Software Engineering Research & Practice, SERP, 2009, 2*, 571–575.
Raza, A., & 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., & Ahmed, F. (2010a). 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, 2010-b.
Raza, A., Capretz, L. F., & Ahmed, F. (2010b). Users' perception of open source usability: An empirical study. Engineering with Computers, Online First, 2011, 1–13.
Raza, A., Capretz, L. F., & Ahmed, F. (2011). An empirical study of open source software usability – The industrial perspective. *International Journal of Open Source Software and Processes, 3*(1), 1–16.
Seffah, A., Donyaee, M., Kline, R., & Padda, H. (2006). Usability measurement and metrics: A consolidated model. *Software Quality Journal, 14*(2), 159–178.
Stevens, J. (1986). *Applied multivariate statistics for the social sciences*. NJ: Hillsdale.
van de Ven, A. H., & Ferry, D. L. (1980). *Measuring and assessing organizations*. NY: John Wiley & Son.
von Eye, A., & Mun, E. Y. (2005). *Analyzing rater agreement manifest variable methods*. London: LEA Publishers.
Wang, Y., & King, G. (2000). *Software engineering processes: Principles and Applications*. NY: CRC Press, pp. 191–219.
Winter, S., Wagner, S., & Deissenboeck, F. (2008). A comprehensive model of usability. In J. Gulliksen, et al. (Eds.), EIS 2007, LNCS 4940 (pp. 106–122).
Zhao, L., Deek, F. P. (2005). Improving open source software usability. 11th Americas conference on information systems (pp. 923–928), August 11–14, Omaha, USA, 2005.
Zhao, L., 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.