Engineering with Computers (2012) 28:109–121
DOI 10.1007/s00366-011-0222-1

ORIGINAL ARTICLE

# Users' perception of open source usability: an empirical study

**Arif Raza · Luiz Fernando Capretz · Faheem Ahmed**

Received: 7 May 2010 / Accepted: 9 May 2011 / Published online: 21 May 2011

**Abstract** The number of open source software (OSS) users has increased in recent years. No longer are they limited to technically adept software developers. Many believe that the OSS market share could increase tremendously provided OSS had systems that were easier to use. Although examples of good usable open source software exist, it is agreed that OSS can be made more usable. This study presents an empirical investigation to study the impact of some key factors on OSS usability from the end users' point of view. The research model studies and establishes the relationship between the key usability factors from the users' perspective and OSS usability. A data set of 102 OSS users from 13 open source projects of various sizes was used to study the research model. The results of this study provide empirical evidence by indicating that the highlighted key factors play a significant role in improving OSS usability.

**Keywords** Usability testing · Software quality · Statistical methods · User issues

## 1 Introduction

Software plays an ever-increasing role in our society. However, software systems often fail to deliver as promised. Usability concerns need to be addressed in many of the software systems that we use every day. Open source software allows its users to use, inspect, modify and distribute it in modified or unmodified form to others [1]. von Krogh and Spaeth [2] propose that such an open license is one of the major reasons for OSS's existence. They believe that to get the best out of open source phenomena, "information systems research should remain open to dialog with other areas and disciplines." Levesque [3] argues that for OSS to be widely accepted, it must address issues like user interface design, documentation, feature-centric development, self-programming and "religious blindness". Polancic et al. [4], however, question the quality of OSS projects. They propose an assessment model to evaluate an OSS product. The empirical study of Paulson et al. [5] supports the belief that defects are found and fixed more rapidly in OSS as compared to closed source software products.

Although it is not presumed that every OSS has a bad interface, it is believed there are issues related to usability in OSS. In the words of Nichols and Twidale [6], "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". Hedberg et al. [7] also identify that due to the constant growth in number of novice/non-technical users of OSS, its usability needs to be improved. According to Çetin and Göktürk [8], OSS usability is a multidimensional problem area particularly due to the fact that usability is not a prime goal of OSS projects; OSS developers are not aware of the importance of usability and users' requirements and there is a lack of interaction between developers and the human computer interaction (HCI) community.

This study contributes to understanding the effects of some key usability factors on OSS usability through an

A. Raza (✉) · L. F. Capretz
Department of Electrical and Computer Engineering, University of Western Ontario, London, ON N6A 5B9, Canada
e-mail: araza7@uwo.ca

F. Ahmed
Faculty of Information Technology, United Arab Emirates University, P. O. Box 17551, Al Ain, United Arab Emirates

empirical investigation. A quantitative survey of OSS users of different OSS projects is conducted and is reported here. The survey is to analyze the conceptual model and the hypotheses of the study. The results provide evidence that the stated key factors play an important role toward OSS usability.

The literature review that motivates this research work is presented in the next section. It also helps in the selection of the key factors for the study. Section 3 illustrates the research model and the hypotheses of this study. Section 4 explains the research methodology, data collection process and the experimental setup in its first part, reliability and validity analysis of the measuring instrument in the second part and data analysis procedures in its third part. In Sect. 5, the hypotheses are tested and the results are analyzed followed by the discussion of results in Sect. 6 that also includes the limitations of the study. Finally, the paper concludes with Sect. 7.

## 2 Literature review

### 2.1 Open source software quality, in general

Porter et al. [9] observe the inconsistency in open source software mainly due to "short feedback loops between users and core developers". They feel that the frequent release of beta versions of the software, although it may satisfy some users, does frustrate many others who would like more stable software. They also identify unsystematic and undocumented software testing in OSS. Yang and Wang consider free and open source software as a reliable fighting force against the monopoly of proprietary software [10]. They believe that its popularity will ultimately reduce users' dependency on proprietary software. Stol et al. [11] categorize OSS as a suitable field for research, in particular for empirical studies due to easy and freely available data through sourceforge.net, freshmeat.com and mailing lists. However, they identify the need to systematically review these empirical analyses and their results. Stol et al. have conducted a systemic review of OSS-related empirical research and have presented their results. Ferenc et al. [12] realize the importance of OSS in the software industry and emphasize the need to measure the quality and reliability of OSS code through the use of proper tools. They present a tool set to extract facts that are used to calculate object-oriented metrics of real-world software. Hedgebeth [13] considers OSS a valuable collaborative source of knowledge management. He highlights different misconceptions about open source such as "security concerns, a lack of a customer support apparatus, and a perceived inability that businesses cannot make a profit using open source technology" and asserts that OSS is here to stay. This fact is now acknowledged even by big commercial software organizations such as Microsoft, IBM, Apple, etc.

Golden et al. [14] have come up with a usability-supporting architectural pattern (USAP) that supports specific usability issues at the architectural level. They observe that usability concerns could be better addressed if "implications of usability heuristics for software design" are made clear and explicit to the software designers. Nakagawa et al. [15] also believe in the direct relationship between software architecture and OSS quality. They point out to OSS developers that architectural knowledge, styles, patterns and evaluation methods should be applied to achieve high quality and success of OSS projects.

Stamelos et al.'s [16] empirical study concludes that user satisfaction is related to the average size of components in an open source application. They also support the idea of OSS having its own quality standards. According to Feller and Fitzgerald [17], OSS users have traditionally been the developers, testers and documenters themselves or the experts in the field resulting in an overlap of developers and users. However, with the entrance of OSS into the mainstream and the interest and support of big corporate players like IBM, Apple and Oracle, non-expert users are attracted to OSS. No cost (generally), high quality and support have been the motivating forces as well.

Analyzing the effects of different development practices on product quality in OSS, Koch and Neumann [18] found a significant relationship between process attributes and product quality. Gyimothy et al. [19] underline the need to study the quality and reliability of OSS due to non-conventional development and management methodologies employed by the OSS community. They have come up with a toolset to calculate metrics from C++ source code of real-world software. Ferenc et al. [12] are curious about the quality and reliability of OSS as it is developed outside the controlled environment and without proper company management. Mansfield-Devine, however, has a different viewpoint. He states, "By monitoring the development of an OSS project, joining mailing lists, seeing how quickly issues are fixed, how often releases are made and so on, users can verify how well the project is being managed" [20].

Samoladas et al. [21] present a hierarchical quality model SQO-OSS to evaluate source code and related processes in open source software. The model supports an automated software evaluation system and is based on calculation of metric values. The authors state, "Our model evaluates all aspects of OSS development, both the product (code) and the community". The evaluation process also provides a profile-based evaluation algorithm.

Golden proposes open source maturity model (OSMM) and relates the OSS quality to its maturity by considering "Product software, Support, Documentation, Training, Product integrations and Professional services" [22]. Each of these factors is evaluated to find an accumulated score. Although OSMM is simple and thus easy to apply, it overlooks some important software parameters, such as the source code.

In the last decade, two well-known methodologies have emerged, namely, Open Business Readiness Rating (OpenBRR) by Carnegie Mellon West and Intel [23] and Qualification and Selection Open Source (QSOS) sponsored by Atos Origin [24]. Deprez and Alexandre have performed a detailed comparative study of these two assessment methodologies on their "overall approaches, their scoring procedures and their evaluation criteria" [25]. However, their comparison is based on the description of the methodologies and not on their empirical application.

Çetin and Göktürk 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."

### 2.2 Usability factors: literature review of concepts

In the ISO 9241-11 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" [26]. The International Organization for Standardization and the International Electro Technical Commission ISO/IEC 9126-1 places software quality attributes into six categories, namely functionality, reliability, usability, efficiency, maintainability and portability [27]. In the standard, usability is defined as "The capability of the software product to be understood, learned, used and attractive to the user, when used under specified conditions."

While examining usability practices in OSS, Nichols and Twidale [28] 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 of 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. Aberdour [29] 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. [7] 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 insure errors are caught before the release of the software.

Sampson criticizes OSS usability by wondering whether OSS developers ever consult a usability expert or explore any of the standards [30]. He refers to the same old 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 [31] while exploring the usability related issues in open source development conclude that "Scarcity of expertise, Bug reporting and classification, and Heterogeneity in usability discussions" are the areas that need to be focused on by the OSS developers. The more that the users' expectations and requirements are addressed in open source software, the more acceptances it will receive. Bodker et al. [32] believe that there is a gap between the OSS "developer-users" and their potential users. They posit that enabling the employees of organizations and institutions to come up with 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 [6] also highlight few related problems in this scenario such as the traditional approach by OSS developers to develop software for an "elite" class of technically adept users. However, 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 need to insure that new users find their products usable and adaptable. It is time for OSS architects, designers and developers to realize that they are not the ultimate users of their applications.

Zhao and Deek [33] highlight the problem of reporting bugs by OSS users. They observe that when an average user wants to report an error, particularly related to usability, s/he does not know how to do it effectively. Observing a similar problem, Nichols and Twidale [28] identify the difficulties faced by the users in reporting usability bugs as "difficulties that a user may experience with a Graphical User Interface may not be easy to describe textually". They observe that there is a bias in treating usability bugs as compared to functionality bugs. Usability issues, as expected, are more subjective in nature and more debatable. As an example, a user interface (UI) element may be more confusing to some people and less to others. Such issues could prolong the discussion of analyzing and fixing usability bugs. Cetin et al. [34] observe that due to "the lack of a suitable usability reporting interface", usability issues are reported less than when they really exist. It is proposed that effective feedback from end users can be one of the ways to improve OSS usability. This can be achieved by providing them with an easy and convenient way to report the errors they encounter while using the software.

Interactive help features in software that addresses users' problems dynamically can be a step toward addressing the usability problem. However to provide help, it is necessary to know exactly what sort of help is actually expected and may be asked for. Furthermore, such help features need to be designed to provide help to all sorts of users, not only to the expert ones. This obviously is not an easy task as Shneiderman [35] highlights that designing software for any expert computer user is difficult as it is, let alone designing for anyone to use. He states that lowering the cost of hardware and computer accessories provides access for more people, but interface and information design has to play its role. Viorres et al. [36] argue that the majority of disabled users prefer to use proprietary software due to better accessibility along with assistive technology, even though OSS claims to have the "right to access for all". While discussing the user's problems, it should not be forgotten that there are users having problems, or the elderly or children who are novices to the computer technology.

Formal usability learning by software architects, designers and developers can be an acknowledgement of the problem as well as a part of the solution. Practice of HCI is commonly referred to as usability engineering. To better understand the user's point of view, software developers need to learn HCI and usability principles. Rusu et al. [37] highlight the importance of HCI education for software professionals. They identify that in computer science (CS) design courses, HCI issues are generally given a secondary level of importance. According to Faulkner and Culwin [38], HCI and software engineering educators are usually in different camps. The authors feel that there is a need for more interaction between HCI and SE. According to them, HCI should be adopted as the underlying principle to systems development. Zhao and Deek [33] however suggest that OSS users need to be trained too so that they are able to do usability inspections in an effective and efficient way.

Hedberg et al. [7] propose the incorporation of usability guidelines, active participation of usability experts in OSS projects, usability testing and bug reporting. In-depth empirical research is needed to understand the challenges related to usability and quality assurance in OSS. Nichols and Twidale [28] refer to Human Interface Guidelines (HIGs) that not only can prevent confusion about usability issues, but also can be considered as an authority on what shall be done. Çetin and Göktürk [8] also identify that there is no consensus of usability guidelines for OSS developers from usability experts. Iivari et al. [39] believe that the growing user population of OSS is more interested in usable systems than in their development. They recommend the involvement of HCI experts in OSS development. Therefore, it is first necessary to understand how usability issues are practiced in OSS environments and then to come up with a method to evaluate and measure usability of OSS projects.

Benson et al. [40] call usable software "a win–win situation for developers, the corporations, and—most importantly—the users". Folmer and Bosch [41] in their survey about usability engineering practices identify that usability issues are typically discovered late in the software development process and hence are expensive to implement. They advocate that "usability should drive design at all stages" and assert that usability issues can best be addressed if incorporated at the architectural design level. Hedberg et al. [7] believe that high quality and usability can be insured in OSS through proven methods and processes; however, there is a need to find ways to adapt to user-centered design methods that can fit in a distributed environment of OSS development.

## 3 Research model and the hypotheses

This work presents a research model for analyzing the relationship between key usability factors and open source software usability, as shown in Fig. 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.

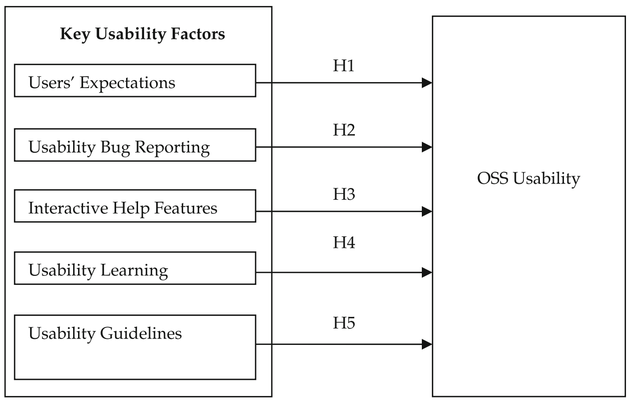

**Fig. 1** Research model

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?"

The multiple linear regression equation of the model is as follows:

$$\text{OSS usability} = f_0 + f_1 v_1 + f_2 v_2 + f_3 v_3 + f_4 v_4 + f_5 v_5 \quad (1)$$

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. To empirically investigate the research question, the five hypotheses are derived as presented below:

**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

## 4 Research methodology

Open source software projects deal with different categories of applications, such as Database, Desktop Environment, Education, Finance, Games/Entertainment and Networking. To collect the data, we sent e-mails 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 presented in Appendix, 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 Fig. 2.

We assured the participants that our survey was confidential and that their identity would not be disclosed. However, 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, 2 respondents categorized themselves as novice computer users, 10 considered themselves as average computer users and 89 of the respondents believed that they were experienced users, as demonstrated in Fig. 3. These statistics indicate the dominance of experienced computer users in the OSS arena.

### 4.1 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 Appendix 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 20 separate items to measure the independent variables and 4 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 was practiced within each project. The items for all five usability factors were labeled sequentially in Appendix and numbered 1 through 20. 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 were labeled sequentially from one through four in Appendix.

### 4.2 Reliability and validity analysis of 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 most commonly used approaches in empirical studies were used to conduct reliability and validity analyses of the measuring instruments of the study. The reliability of the multiple-item measurement scales of the five usability factors was evaluated using an internal-consistency analysis, which was performed using coefficient alpha [42]. In our analysis, the coefficient alpha ranged from 0.56 to 0.60 as shown in Table 1 and Fig. 4. van de Ven and Ferry [43] stated that a reliability coefficient of 0.55 or higher was satisfactory, and Osterhof [44] suggested that 0.60 or higher was satisfactory. Therefore, it was concluded that the variable items developed for this empirical investigation were reliable.

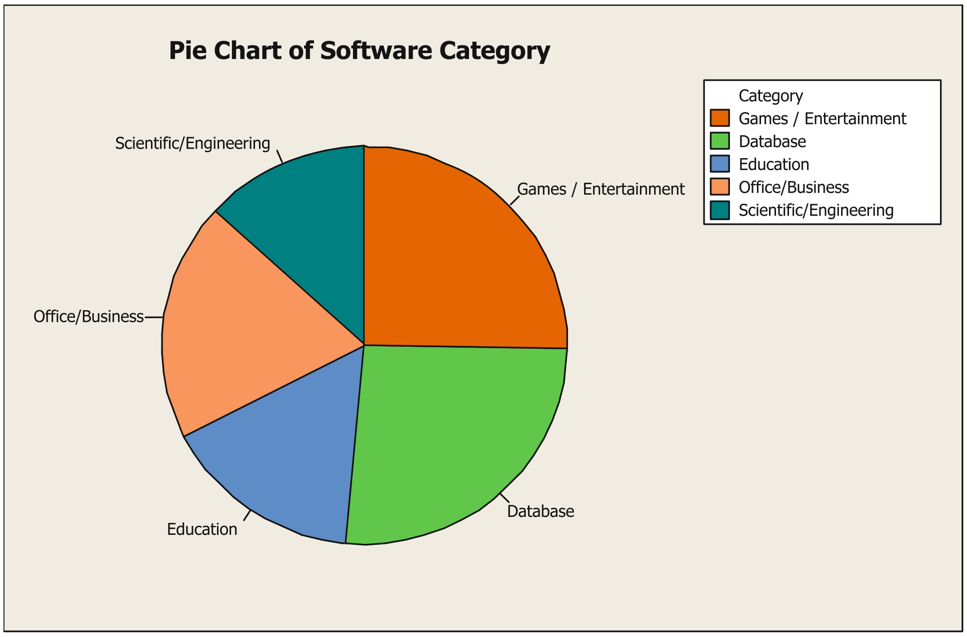

**Fig. 2** Respondents' distribution

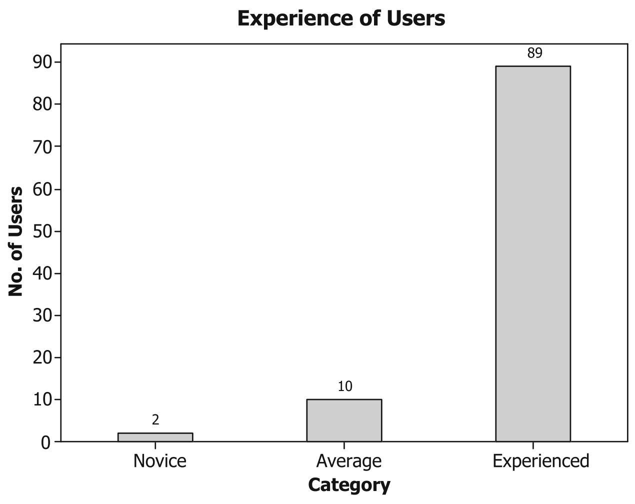

**Fig. 3** Experience of users

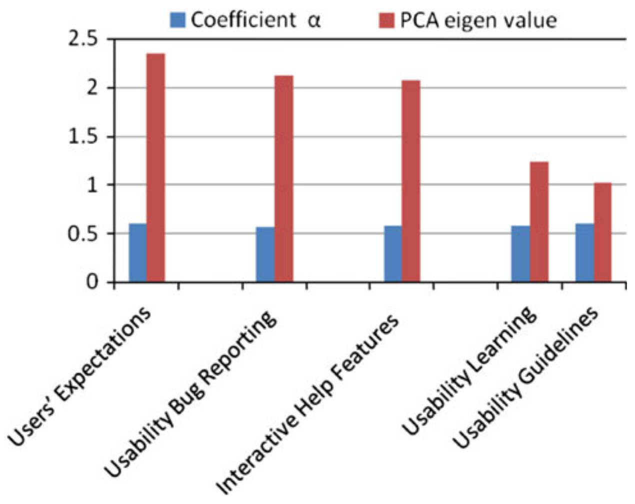

**Fig. 4** Coefficient alpha and principal component analysis of variables (users' perspective)

**Table 1** Coefficient alpha and principal component analysis (PCA) of variables

| Usability factors | Item no. | Coefficient α | PCA eigenvalue |
|---|---|---|---|
| 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 [45], convergent validity occurs in a given assembly when the scale items are correlated and move in the same direction. The principal component analysis was performed for all five usability factors [46], and is reported in Table 1 and Fig. 4. The eigenvalue from Kaiser was used as a reference point to observe the construct validity using principal component analysis [47]. We used eigenvalue-one-criterion, also known as Kaiser Criterion [48, 49] here, which means any component having an eigenvalue greater than one is retained. Eigenvalue analysis revealed that all five variables completely formed a single factor. Therefore, it is concluded that the convergent validity can be regarded as sufficient.

### 4.3 Data analysis procedure

We analyzed the research model and the significance of hypotheses H1–H5 using different statistical techniques in three phases. In Phase I we used normal distribution tests and parametric statistics, whereas in Phase II we used non-parametric statistics. Due to the relatively small sample size, both parametric as well as non-parametric statistical approaches were used to reduce the threat to external validity. Since our measuring instruments had multiple items for all five independent variables as well as the dependent variable (refer to Appendix), their respondents' ratings were summed to get a composite value for each of them. Tests were conducted for hypotheses H1–H5 using parametric statistics to determine the Pearson correlation coefficient. For non-parametric statistics, Spearman correlation coefficient tests were determined for hypotheses H1–H5. To deal with the limitations of relatively small sample size and to increase the reliability of the results, hypotheses H1–H5 of the research model were tested using the partial least square (PLS) technique in Phase III. According to Fornell and Bookstein and Joreskog and Wold, the PLS technique is helpful in dealing with issues such as complexity, non-normal distribution, low theoretical information and small sample size [50, 51]. The statistical calculations were performed using minitab-15.

## 5 Hypotheses testing and results

### 5.1 Phase I

To test hypotheses H1–H5 of the research model, as shown in Fig. 1, parametric statistics were 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 2 and Fig. 5. "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 hypothesis is true, and consequently the more "significant" the result is, in the sense of statistical significance" [52].

The Pearson correlation coefficient between the users' expectations and OSS usability was found to be positive (0.221) at $P < 0.05$, and hence justified the hypothesis H1. A Pearson correlation coefficient of 0.367 was observed at $P < 0.05$ between usability bug reporting and OSS usability and hence was found to be significant. The hypothesis H3 was accepted based on the Pearson correlation coefficient (0.224) at $P < 0.05$, between the interactive help features and OSS usability. The positive correlation coefficient of 0.497 at $P < 0.05$ was also observed between the OSS usability and usability learning, which meant that H4 was accepted. However, hypothesis H5 was found to be insignificant after analyzing the Pearson correlation coefficient of 0.109 at $P = 0.275$ between usability guidelines and OSS usability. Therefore, hypothesis H5 that deals with usability guidelines and OSS usability was rejected. Hence, as observed and reported above, hypotheses H1, H2, H3 and H4 were found to be statistically significant and were accepted, whereas H5 was not supported and therefore was rejected.

### 5.2 Phase II

Non-parametric statistical testing was conducted in this phase by examining the Spearman correlation coefficient between individual independent variables (usability factors) and the dependent variable (OSS usability). The results of the statistical calculations for the Spearman correlation coefficient are also displayed in Table 2 and Fig. 5. The Spearman correlation coefficient between the users' expectations and OSS usability was found to be positive (0.281) at $P < 0.05$, and hence justified the hypothesis H1. The Spearman correlation coefficient of 0.371 in hypothesis H2 was observed at $P < 0.05$ between usability bug reporting and OSS usability and hence was significant. The hypothesis H3 was accepted based on the

**Table 2** Hypotheses testing using parametric and non-parametric correlation coefficients

| Hypothesis | Usability factor | Pearson correlation coefficient | Spearman correlation coefficient |
|---|---|---|---|
| H1 | Users' expectations | 0.221* | 0.281* |
| H2 | Usability bug reporting | 0.367* | 0.371* |
| H3 | Interactive help features | 0.224* | 0.271* |
| H4 | Usability learning | 0.497* | 0.422* |
| H5 | Usability guidelines | 0.109** | 0.025** |

* Significant at $P < 0.05$

** Insignificant at $P > 0.05$

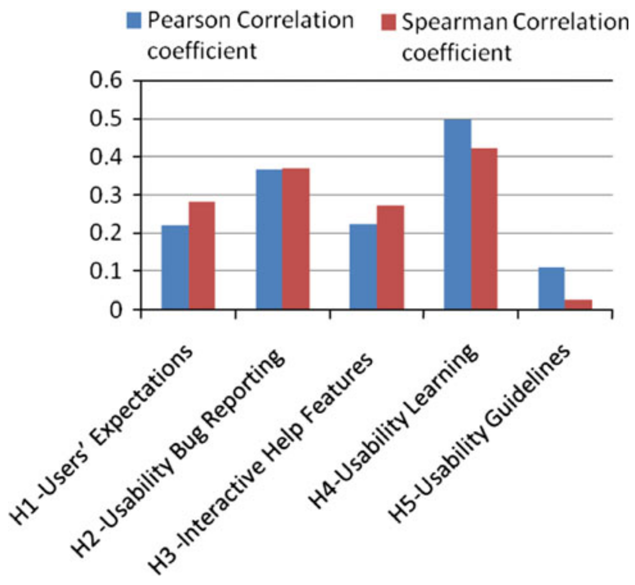

**Fig. 5** Hypotheses testing (users' perspective)

**Table 3** Hypotheses testing using partial least square (PLS) regression

| Hypothesis | Usability factor | Path coefficient | $R^2$ | $F$ ratio |
|---|---|---|---|---|
| H1 | Users' expectations | 0.221 | 0.049 | 5.15* |
| H2 | Usability bug reporting | 0.584 | 0.135 | 15.57* |
| H3 | Interactive help features | 0.463 | 0.050 | 5.26* |
| H4 | Usability learning | 0.619 | 0.247 | 32.82* |
| H5 | Usability guidelines | 0.217 | 0.012 | 1.20** |

* Significant at $P < 0.05$

** Insignificant at $P > 0.05$

Spearman correlation coefficient (0.271) at $P < 0.05$, between the interactive help features and OSS usability. The positive Spearman correlation coefficient of 0.422 at $P < 0.05$ was also observed between the OSS usability and usability learning, which meant that H4 was accepted as well. However, for hypothesis H5 the Spearman correlation coefficient of 0.025 was observed at $P = 0.799$ between the usability guidelines and OSS usability. As no significant relationship was found between the usability guidelines and OSS usability in this test at $P < 0.05$, hypothesis H5 was rejected.

Hence, as observed and presented above, H1, H2, H3 and H4 were found to be statistically significant and accepted, whereas H5 was not supported and hence rejected in the non-parametric analysis as well.

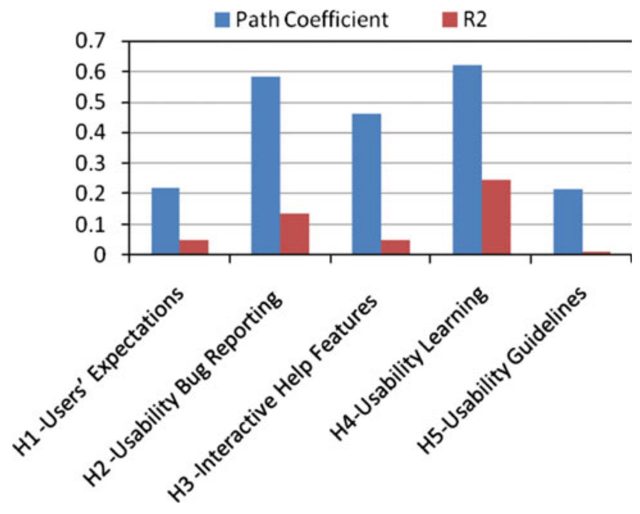

**Fig. 6** Hypotheses testing using PLS regression (users' perspective)

### 5.3 Phase III

To perform the cross validation of the results obtained in Phase I and Phase II, the partial least square (PLS) technique was used in this phase of hypotheses testing. The direction and significance of hypotheses H1–H5 were examined. In PLS, the dependent variable of the research model, i.e., OSS usability was used as the response variable and independent key usability factors as predicates. The test results containing observed values of path coefficient, $R^2$ and $F$ ratio are shown in Table 3 and Fig. 6. The users' expectations were observed to be significant at $P < 0.05$ with path coefficient 0.221, $R^2$: 0.049 and $F$ ratio 5.15. Usability bug reporting had a path coefficient of 0.584 with $R^2$: 0.135 and $F$ ratio 15.57 and was found to be significant at $P < 0.05$. Interactive help features were observed to have the same direction as those proposed in hypothesis H3 with path coefficient 0.463, $R^2$: 0.050 and $F$ ratio 5.26 at $P < 0.05$. Usability learning was also found to be in conformance with hypothesis H4 with observed values of path coefficient 0.619, $R^2$: 0.247 and $F$ ratio 32.82 at $P < 0.05$. Finally, the usability guidelines were found to have path coefficient 0.217, $R^2$: 0.012 and $F$ ratio 1.20 with observed $P = 0.275$. Hence, in this phase, as in Phase I and II, hypothesis H5 that deals with usability guidelines and OSS usability was not found to be statistically significant at $P < 0.05$ and hence was rejected.

### 5.4 Testing of the research model

The multiple linear regression equation of our research model is depicted by Eq. 1. The testing process included the regression analysis, which yielded the values of the model coefficients and their direction of association. OSS usability was used as response variable and the usability factors as predicators. Table 4 displays the regression analysis results of the research model. The path coefficient of all five variables was found to be positive, whereas the *t* statistics of four out of five variables, namely users' expectations, usability bug reporting, interactive help features and usability learning, were found to be statistically significant at $P < 0.05$. The *t* value of the usability guidelines was observed as 0.40 at $P = 0.688$, thus making usability guidelines statistically insignificant in this research model. $R^2$ and adjusted $R^2$ of the overall research model were observed as 0.322 and 0.286 with an *F* ratio of 9.10 significant at $P < 0.05$.

Recapping Eq. 1 by inserting the model coefficient values, we get:

$$\begin{aligned}\text{OSS usability} &= 3.92 + 0.22v_1 + 0.20v_2 + 0.19v_3 \\ &\quad + 0.38v_4 + 0.04v_5\end{aligned}$$

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

## 6 Discussion of the results

The emerging use of open source software in recent years is due to reasons such as easy (and mostly free) access and availability of the Internet. However, many believe that post-release maintenance, quality control and management are among those areas where closed proprietary software has the upper hand. Twidale stresses the need for "participatory usability involving the coordination of end users and developers" [53]. With the popularity of OSS among organizations as well as common novice users, the OSS community is no longer limited to "technically adept" people alone. Hence, the requirements and expectations are not the same as they were a decade ago, when only software developers were supposed to be the only OSS users. Through empirical investigation this research highlights the relationship of key factors in the present research model and the OSS usability process from actual OSS users' points of view.

In their empirical study to measure success of OSS projects, Lee et al. [54] conclude that "OSS use was significantly influenced by software quality and user satisfaction". We believe that to achieve user satisfaction, software designers and developers need to understand their expectations and requirements. Furthermore, software developers need to communicate more with the target audience of their product to better understand their perspective [55]. In our survey, 41% of the respondents agreed, 29% remained neutral and the rest 30% disagreed with the statement, "User interface of OSS should follow standards and norms of proprietary software to make them easy to use,". On the other hand, only 17% agreed with the statement, "I believe OSS is not meant for novice non-technical users," 5% remained neutral and 78% disagreed. About the statement, "Formal feedback from users is missing and thus needed in an OSS environment," 36% agreed, 31% remained neutral and the rest chose to disagree with the statement. For the last statement related to "Users' Expectations", which was, "Proprietary software addresses users' expectations and requirements better than OSS," 17% agreed, 27% remained neutral and the rest disagreed. We have found in the empirical investigation a positive relationship between users' expectations and the OSS usability. "Users' Expectations" could thus be taken by OSS developers community to be a key issue necessary to improve usability of their projects.

A subjective matter like software usability cannot be directly measured. Users also find it difficult to report such errors. Nichols and Twidale [28] identify that it is not easy for a user to describe and hence report the difficulties s/he faces in software Graphical User Interface (GUI). Software designers and developers need to make it convenient for their users to report usability related bugs. Of the respondents in our survey, 78% agreed that the ease of reporting errors in software would increase their level of satisfaction; 90% of the respondents did not agree with the statement "I never report an error, so usability bug reporting does not affect me." However, only 19% agreed, 42% remained neutral and the rest disagreed with our survey statement, "For effective usability inspections, user training is required." Moreover, 63% agreed that, "Usability bugs reflect users' expectations; therefore, they need to be fixed on priority." Consequently, our empirical analysis also confirms a positive association between usability bug reporting and OSS usability. Thus, "usability bug reporting and fixing" is considered to have a positive impact on OSS usability.

While discussing software users, their expectations and problems, the fact should not be overlooked that users do include elderly, children and people having some

**Table 4** Multiple linear regression analysis of the research model

| Model coefficient name | Model coefficient | Coefficient value | *t* value |
|---|---|---|---|
| Users' expectations | $f_1$ | 0.220 | 2.55* |
| Usability bug reporting | $f_2$ | 0.199 | 1.79* |
| Interactive help features | $f_3$ | 0.188 | 1.69* |
| Usability learning | $f_4$ | 0.382 | 4.40* |
| Usability guidelines | $f_5$ | 0.036 | 0.40** |
| Constant | $f_0$ | 3.92 | 0.45* |

* Significant at $P < 0.05$

** Insignificant at $P > 0.05$

disability. Thus, to increase the acceptance level of OSS among different categories of users, help features of software need to be made interactive and dynamic. Of our respondents, 66% believed that "interactive help would increase ease-of-use of open source software." However, 68% did not agree with the statement, "A novice user needs only basic features of software so interactive help features would not have much impact." Moreover, 71% of the respondents of our survey agreed that interactive help would increase their learnability of open source software. Furthermore, our empirical investigation also supports our hypothesis, "Interactive help features in software have a positive impact on usability in OSS."

Benson et al. [40] observed that quite often OSS was "created by engineers for engineers". Hence, to have a better market share and accessibility, they emphasized that OSS developers need to make suitable usability tactics their top priority. OSS managers and developers thus need to consider the importance of usable systems more seriously. Of our survey's respondents, 79% agreed that OSS developers must learn how to incorporate users' requirements and usability aspects in their software designs; 40% agreed that "lack of usability knowledge is the main cause of poor usability of OSS systems" and 54% of the respondents believed that "the realization of the fact that a software system is for end users is more important than the formal usability learning." Finally, 36% agreed, 50% remained neutral, and 13% of 102 respondents disagreed with the statement, "OSS developers should use quantifiable usability metrics to measure it objectively and effectively." Furthermore, our empirical analyses confirmed the positive relationship between usability learning and OSS usability as well. Usability learning was thus determined to be one of the key issues needed to improve the usability of OSS projects.

According to Hedberg et al. [7], if existing and already proven methods are adopted, high quality and usability could be insured in OSS. As already mentioned above, Nichols and Twidale [28] advocate that Human Interface Guidelines (HIGs) have a consensus about usability issues in terms of an authority on what shall be done. Of the respondents, 35% felt that "there should be a standardized user interface and usability guidelines that OSS developers should follow in their designs"; 41%, however, disagreed with the statement and the rest remained neutral. As much as 67% were of the opinion that "usability guidelines could act as a standard and checklist against which software could be inspected." Similarly, 63% of the respondents in our survey felt that the strict implementation of usability guidelines takes away the OSS developer's freedom and 27% agreed that "standardized usability guidelines are impracticable in an OSS environment." Considering our empirical analysis, in the parametric and non-parametric statistical analysis as well as in PLS and multiple regression testing, the results did not support the positive relationship between usability guidelines and OSS usability (refer to Tables 2, 3, 4 above). Therefore, it is concluded that our study was not able to prove a positive association of usability guidelines and OSS usability.

To assess the dependent variable, OSS usability, we presented four statements to our respondents as well. As much as 52% of respondents of our survey agreed with the statement, "OSS with standardized usability features will help users to compare the usability of different software" and 60% believed that "poor usability being the major hurdle, improved usable OSS systems will result in switching users from proprietary software to OSS." Fifty-nine percent were of the opinion that "usable software with satisfied users guarantees its success" and 81% of OSS users of our survey showed their agreement with the statement, "open source software having improved usability and adaptability for less technical and novice users will end up benefiting all users," thus indicating their strong support toward improved usability in OSS environment.

## 6.1 Limitations of the study and threats to external validity

Like any empirical investigation, this study has certain limitations. The researcher's ability to generalize the experimental outcome to industrial practice is generally limited by threats to external validity [56], which was the case with this study as well. We took specific measures to support external validity such as using a random sampling technique to select respondents from the population to conduct experiments. We retrieved the data from the most active and well-known OSS reporting Web site, sourceforge.net, which has a large number of projects listed. The increased popularity of empirical methodology in software engineering has also raised concerns of an ethical nature [57, 58]. The recommended ethical principles were followed to insure that the empirical investigation conducted would not violate any form of recommended experimental ethics. Another aspect of validity concerned whether or not the study results reported corresponded to previous findings. First of all, there is the selection of independent variables in this work. We have used five independent variables to relate to the dependent variable of OSS usability. Of course, there could be other key factors that influence usability, but the scope of this study was kept within the area of open source software and end users' point of view. Some other contributing factors like users' educational and cultural background, and their experience in using OSS projects in comparison to closed proprietary software projects developed in big organizations have not been considered in this study. It is worth mentioning here that we are also conducting three more studies, on the same format, to study the perspective of developers, contributors and industries on open source software usability. Consequently, many more usability factors will also be empirically investigated.

Another limitation of this study was its relatively small sample size. Although we sent our survey to a large number of OSS users who subscribed to different (13) categories of software, we received only 102 responses. The relatively small sample size in terms of number of respondents was a potential threat to the external validity of this study. We followed appropriate research procedures by conducting and reporting tests to improve the reliability and validity of the study, and certain measures were also taken to insure the external validity.

## 7 Conclusion

This study investigates the effects of key factors on OSS usability from the end user's perspective. Results of the empirical analysis show that the stated key factors of our research model assist in the improvement of OSS usability. The results support the hypotheses that users' expectations, usability bug reporting and fixing, interactive help features and usability learning have a positive impact on usability of OSS projects. However, we could not find any significant statistical support for usability guidelines on OSS usability. The study conducted and reported here can assist OSS designers and developers to better comprehend the efficacy of the relationships of the stated key factors and usability of their projects. The stated usability factors may be considered by the OSS development community to address usability issues of their projects. This empirical investigation provided some justification for us to consider these key factors as measuring instruments for a maturity model to assess the usability of open source software project, which we have recently developed.

## Appendix: Key usability factors from OSS user's point of view (measuring instrument)

Users' expectations

1. User interface of OSS should follow standards and norms of proprietary software to make them 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.

Usability bug reporting and fixing

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. For effective usability inspections, user training is required.
8. Usability bugs reflect users' expectations; therefore, they need to be fixed on priority.

Interactive help features

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

Usability learning

13. OSS developers must learn how to incorporate users' requirements and usability aspects into their software designs.
14. Lack of usability knowledge is the main cause of poor usability of OSS systems.
15. The realization of the fact that a software system is for end users (not for the developers themselves) is more important than the formal usability learning.
16. OSS developers should use quantifiable usability metrics to measure it objectively and effectively.

Usability guidelines for OSS developers

17. There should be a standardized user interface and usability guidelines that OSS developers should follow in their designs.
18. Usability guidelines could act as a standard and checklist against which software could be inspected.
19. Strict implementation of usability guidelines will take away the OSS developer's freedom.
20. Standardized usability guidelines are impracticable in an OSS environment.

OSS usability

1. OSS with standardized usability features will help users to compare the usability of different software.
2. Poor usability being the major hurdle, improved usable OSS systems will result in switching users from proprietary software to OSS.
3. Usable software with satisfied users guarantees its success.
4. Open source software having improved usability and adaptability for less technical and novice users will end up benefiting all users.

## References

1. Raymond ES (1999) The cathedral and the bazaar. O'Reilly, Sebastopol
2. von Krogh G, Spaeth S (2007) The open source software phenomenon: characteristics that promote research. J Strateg Inform Syst 16(3):236–253
3. Levesque M (2004) Fundamental issues with open source software development. First Monday 9(4)
4. Polancic G, Horvat RV, Rozman T (2004) Comparative assessment of open source software using easy accessible data. In: Proceedings of the 26th International Conference on Information Technology Interfaces, ITI 2004 (IEEE Cat. No.04EX794), vol 1, pp 673–8
5. Paulson JW, Succi G, Eberlein A (2004) An empirical study of open-source and closed-source software products. IEEE Trans Softw Eng 30(4):246–256
6. Nichols DM, Twidale MB (2005) The usability of open source software. First Monday 8(1). http://firstmonday.org/issues/issue81/nichols/
7. Hedberg H, Iivari N, Rajanen M, Harjumaa L (2007) Assuring quality and usability in open source software development. In: Proceedings of the first international workshop on emerging trends in floss research and development, FLOSS. IEEE Computer Society, Washington, DC, pp 20–26
8. Çetin G, Göktürk M (2008) A measurement based framework for assessment of usability-centricness of open source software projects. In: Proceedings of IEEE international conference on signal image technology and internet based systems, SITIS '08
9. Porter A, Yilmaz C, Memon AM, Krishna AS, Schmidt DC, Gokhale A (2006) Techniques and processes for improving the quality and performance of open-source software. Softw Process Improv Pract 11(2):163–176
10. Yang J, Wang J (2008) Review on free and open source software. In: Proceedings of the 2008 IEEE international conference on service operations and logistics, and informatics (SOLI), pp 1044–1049
11. Stol KJ, Babar MA, Russo B, Fitzgerald B (2009) The use of empirical methods in open source software research: facts, trends and future directions. In: 2009 ICSE workshop on emerging trends in free/libre/open source software research and development (FLOSS 2009), pp 19–24
12. Ferenc R, Siket I, Gyimothy T (2004) Extracting facts from open source software. In: Proceedings of the 20th IEEE international conference on software maintenance, pp 60–69
13. Hedgebeth D (2007) Gaining competitive advantage in a knowledge-based economy through the utilization of open source software. VINE J Inform Knowl Manag Syst 37(3):284–294
14. Golden E, John BE, Bass L (2005) The value of a usability-supporting architectural pattern in software architecture design: a controlled experiment. In: Proceedings of the 27th international conference on software engineering, 15–21 May, St. Louis, MO, USA
15. Nakagawa EY, de Sousa EPM, de Brito MK, de Faria AG, Morelli LB, Maldonado JC (2008) Software architecture relevance in open source software evolution: a case study. In: Proceedings of the 2008 IEEE 32nd international computer software and applications conference (COMPSAC), pp 1234–1239
16. Stamelos L, Angelis L, Oikonomou A, Bleris GL (2002) Code quality analysis in open source software development. Inform Syst J 12(1):43–60
17. Feller J, Fitzgerald B (2000) A Framework Analysis of the Open Source Software Development Paradigm. In: Proceedings of the 21st annual international conference on information systems, Brisbane, Queensland, Australia, pp 58–69
18. Koch S, Neumann C (2008) Exploring the effects of process characteristics on product quality in open source software development. J Database Manag 19(2):31–57
19. Gyimothy T, Ferenc R, Siket I (2005) Empirical validation of object-oriented metrics on open source software for fault prediction. IEEE Trans Softw Eng 31(10):897–910
20. Mansfield-Devine S (2008) Open source: does transparency lead to security? Comput Fraud Secur 9:11–13
21. Samoladas I, Gousios G, Spinellis D, Stamelos I (2008) The SQO-OSS quality model: measurement based open source software evaluation. In: Open source development, communities and quality—OSS 2008: 4th international conference on open source systems, pp 237–248
22. Golden B (2005) Making open source ready for the enterprise, the open source maturity model., Extracted from succeeding with open sourceAddison-Wesley Publishing Company, Reading
23. Business Readiness Rating. Business readiness rating for open source. http://www.openbrr.org
24. QSOS. Method for qualification and selection of open source software (qsos) version 1.6. http://www.qsos.org
25. Deprez JC, Alexandre S (2008) Comparing assessment methodologies for free/open source software: OpenBRR and QSOS. In: Jedlitschka A, Salo O (eds) PROFES 2008. LNCS, vol 5089, pp 189–203. Springer, Heidelberg
26. ISO 9241 (1997). Ergonomics requirements for office with visual display terminals (VDTs)
27. International Standard ISO/IEC 9126-1 (2001) Software Engineering—Product Quality—Part 1: Quality Model, 1st edn, pp 9–10
28. Nichols DM, Twidale MB (2006) Usability processes in open source projects. Softw Process Improv Pract 11(2):149–162
29. Aberdour M (2007) Achieving quality in open-source software. IEEE Softw 24(1):58–64. doi:10.1109/MS.2007.2
30. Sampson F (2007) Who said "usability is free"? Interactions 14(4):10–11
31. Twidale MB, Nichols DM (2005) Exploring usability discussions in open source development. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, pp 198–208
32. Bodker M, Nielsen L, Orngreen RN (2007) Enabling user centered design processes in open source communities: usability and internationalization, HCI and culture. In: Proceedings of the Second International Conference on Usability and Internationalization, UI-HCII 2007. Held as Part of HCI International 2007, part I, pp 10–18

33. Zhao L, Deek FP (2006) Exploratory inspection—a learning model for improving open source software usability. In: Proceedings of the Conference on Human Factors in Computing Systems CHI '06, pp 1589–1594
34. Cetin G, Verzulli D, Frings S (2007) An analysis of involvement of HCI experts in distributed software development: practical issues", Online Communities and Social Computing. In: Proceedings of the Second International Conference, OCSC 2007. Held as Part of HCI International 2007, pp 32–40
35. Shneiderman B (2000) Universal usability. Commun ACM 43(5):84–91
36. Viorres N, Xenofon P, Stavrakis M, Vlachogiannis E, Koutsabasis P, Darzentas J (2007) Major HCI challenges for open source software adoption and development. In: Douglas (ed) Proceedings of Schuler. OCSC 2007—Online Communities and Social Computing, Second International Conference, Beijing, China, pp 455–464
37. Rusu C, Rusu V, Roncagliolo S (2008) Usability practice: the appealing way to HCI. In: Proceedings of first international conference on advances in computer–human interaction, achi-2008, pp 265–270
38. Faulkner X, Culwin F (2000) Integrating HCI and SE. ACM SIGCSE Bull 32(3):61–64
39. Iivari N, Hedberg H, Kirves T (2008) Usability in company open source software context. Initial findings from an empirical case study. In: Proceedings of the 4th international conference on open source systems (co-located with the World Computer Congress), Milan
40. Benson C, Muller-Prove M, Mzourek J (2004) Professional usability in open source projects: GNOME, OpenOffice.org, NetBeans. In: Proceedings of the CHI '04 extended abstracts on human factors in computing systems, Vienna, Austria
41. Folmer E, Bosch J (2004) Architecting for usability: a survey. J Syst Softw 70:61–78
42. Cronbach LJ (1951) Coefficient alpha and the internal consistency of tests. Psychometrica 16:297–334
43. van de Ven AH, Ferry DL (1980) Measuring and assessing organizations. Wiley, NY
44. Osterhof A (2001) Classroom applications of educational measurement. Prentice Hall, NJ
45. Campbell DT, Fiske DW (1959) Convergent and discriminant validation by the multi-trait multi-method matrix. Psychol Bull 56:81–105
46. Comrey AL, Lee HB (1992) A first course on factor analysis, 2nd edn. Lawrence Erlbaum, Hillsdale
47. Kaiser HF (1970) A second generation little jiffy. Psychometrika 35:401–417
48. Kaiser HF (1960) The application of electronic computers to factor analysis. Educ Psychol Measur 20:141–151
49. Stevens J (1986) Applied multivariate statistics for the social sciences. Erlbaum, Hillsdale
50. Fornell C, Bookstein FL (1982) Two structural equation models: LISREL and PLS applied to consumer exit voice theory. J Mark Res 19:440–452
51. Joreskog K, Wold H (1982) Systems under indirect observation: causality, structure and prediction. North Holland, The Netherlands
52. Wikipedia (2010). http://en.wikipedia.org/wiki/P-value
53. Twidale M (2005) Silver bullet or fool's gold: supporting usability in open source software development. In: Proceedings of the 27th international conference on software engineering
54. Lee SYT, Kim HW, Gupta S (2009) Measuring open source software success. Omega 37(2):426–438
55. Koppelman H, Van Dijk B (2006) Creating a realistic context for team projects in HCI. SIGCSE Bull 38(3):58–62
56. Wohlin C, Runeson P, Host M, Ohlsson MC, Regnell B, Wesslen A (2000) Experimentation in software engineering. Kluwer Academic Publishers, Norwell
57. Faden RR, Beauchamp TL, King NMP (1986) A history and theory of informed consent. Oxford University Press, New York
58. Katz J (1972) Experimentation with human beings. Russell Sage Foundation, New York