ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2006, Vol.35, No.3A

USABILITY IN OPEN SOURCE SOFTWARE DEVELOPMENT: OPINIONS AND PRACTICE

Morten Sieker Andreasen, Henrik Villemann Nielsen, Simon Ormholt Schrøder, Jan Stage

Department of Computer Science
Aalborg University, Denmark

Abstract. Open Source Software (OSS) development has gained significant importance in the production of software products. Open Source Software developers have produced systems with a functionality that is competitive with similar proprietary software developed by commercial software organizations. Yet OSS is usually designed for and by power-users, and OSS products have been criticized for having little or no emphasis on usability. We have conducted an empirical study of the developers' opinions about usability and the way usability engineering is practiced in a variety of OSS projects. The study included a questionnaire survey and a series of interviews, where we interviewed OSS contributors with both technical and usability backgrounds. Overall we found that OSS developers are interested in usability, but in practice it is not top priority, and OSS projects rarely employs systematic usability evaluation. Most of the efforts are based on common sense. Most developers have a very limited understanding of usability, and there is a lack of resources and evaluation methods fitting into the OSS paradigm.

1. Introduction

Open Source Software (OSS) refers to software released under a license that allows the end user to freely use, distribute and modify the source code of the program. OSS is often described as 'free' software, which reflects the liberty, not the price of the software [0, 0]. Famous OSS projects include Linux, Eclipse, Apache, Tomcat and the Mozilla suite of programs. The majority of OSS projects have been conducted by individuals or small development teams with little formal organization. Yet companies like IBM, SUN and Novell are increasing their involvement with OSS [0].

The classical OSS contributors work in their spare time, solving their dedicated tasks in the project, and communication between them is handled by electronic means as they are usually distributed around the world. The stereotype of OSS developers is that they are the "cowboys of the software world with few formal procedures, actively hostile to any authority other than the hacker ethic" [0]. Nevertheless, it has been emphasized that OSS development results in increased security and quality, since the code is exposed to extreme scrutiny. Thereby, problems are being identified and solved swiftly [0]. In addition, the release cycles of OSS development projects are short and allow rapid software development. There has been an increased attention towards the technological achievements of OSS [0], and it has been concluded that the OSS community has produced software products that are competitive with commercial alternatives [0, 0].

Despite the technological successes, OSS development also faces a number of fundamental challenges. OSS systems have been criticized for a severe lack of usability for non-technical users compared to commercial software [0, 0, 0, 0, 0]. In order to change this, the core problems and the reasons for this deficiency need to be identified and resolved. Yet there has been little research on the reasons why OSS generally does not have the same degree of usability as commercial software and if this situation is likely to change.

In this paper we present results from an empirical study of OSS developers' opinions about usability and the way usability engineering is practiced in a variety of contemporary OSS projects. The aim of this study was to understand current practices and obstacles to change. In the study, we chose to focus on projects carried out by small groups of volunteers. In section 2 we provide an overview of existing research on usability engineering in OSS development. Section 3 describes the methods used in the empirical study. Section 4 presents the main results of the empirical study. In section 5, we discuss three overall themes related to OSS that emerged in the study. Finally, section 6 provides the conclusion.

2. Related Work

OSS has been the subject of several studies. The OSS development paradigm has been described in general, and based on this it was suggested that the OSS development model could potentially solve some of the software industry's problems regarding reliability, pace of development and cost [0]. Another study constructed a basic overview of issues of OSS models for development and distribution of computer documentation, and compared two different methods of implementing open source models for computer documentation [0]. A quantitative study of problem management and bug reporting methods within OSS extracted large amounts of data from the Bugzilla bug reporting tool and analyzed it in order to describe the process that a problem went through from the initial report to it's resolution [0]. This was followed up by an investigation of the relations between bug reports, so called bug reporting networks, and an analysis of a single social negotiation [0, 0].

The majority of research on OSS development, including that mentioned above, is focused on the technical problems. However, there are exceptions to this. The Boston Consulting Group conducted a large, quantitative study of the background and interests of OSS contributors. This revealed the demographics, interests and motivations of a large sample of respondents [0].

The role of user-centered approaches and usability engineering has largely been neglected in research on OSS development. One exception is a study of Netscape's OSS user interface development. When Netscape released its web-browser that was developed as OSS in the Mozilla project, they realized that the design activity could not be skipped without causing severe problems. It also became clear that conventional user-centered approaches like identification of target users should have been employed, because OSS contributors often work with their own use in mind [0]. This is supported by results from a study of communication between developers who used a bug reporting utility in an OSS project. It was concluded that existing human-computer interaction techniques could be used to leverage distributed networked communities of developers and users to address issues of usability [0].

The reasons for the limited focus on users and usability have been discussed. It has been argued that a main obstacle for increasing the focus on users in OSS development is that the vast majority of contributors are software developers with programming backgrounds that do not understand the problems of ordinary users [0]. Others have emphasized lack of competence and resources as the main obstacle. A study of the general usability issues and challenges that OSS was facing concluded that the lack of usability expertise and resources in the OSS community was the key problem [0]. Eric Raymond, one of the founders of the OSS community, has supported this by stating that the OSS community needs "a big player with a lot of money, which is doing systematic user interface end user testing. We're not very good at that yet, we need to find a way to be good at it" [0]. According to Raymond, the reason is the competence needed to organize usability evaluations and the costs of conducting them.

3. Method

We have conducted an empirical study where we aimed to identify the current practice of usability evaluation methods in OSS projects, to understand the opinions and beliefs about usability held by OSS contributors and to clarify the thoughts and experiences of usability evaluators working on OSS projects. The study included three elements: (1) an online questionnaire survey answered by contributors to a variety of OSS projects, (2) interviews with three OSS developers and (3) interviews with five usability evaluators for OSS projects. In this section we describe the research method employed in each of the three elements of the empirical study.

3.1. Questionnaire

The questionnaire survey was conducted from mid September to mid October 2005. The survey consisted of three parts, corresponding to focus areas we wanted to explore: 'About your current project', 'Communication' and 'Usability'. The questions were mixed between quantitative scales where it was possible to choose from a set of predefined options, and qualitative questions where the respondents could answer freely.

Participants: We wanted a high number of respondents. Therefore, we contacted fourteen different OSS projects, where the number of contributors varied from one to more than fifty contributors. We used two main criteria for selecting OSS projects to contact: (1) they should not be developed by professionals and (2) the product should be targeted at mainstream users. This resulted in a total of twenty-four respondents located in fourteen different countries, see Table 1. Via mailings lists and websites related to the projects, we found that the minimum number of contributors that received the invitation to participate in the questionnaire was 293.

Data collection: The questionnaire was available online. We used the PHP/MySQL based survey tool UCCASS. This made it simple to extract the data in a variety of data formats, once the survey had been completed.

Data Analysis: We mapped the quantitative data in diagrams and graphs. All the qualitative data was organized in sections reflecting the three focus areas of the questionnaire, see Figure 1. We used meaning condensation to identify the essential parts of the answers [0]. We used the data to determine tendencies and establish topics for the follow-up interviews with the OSS developers.

Table 1. Overview of OSS projects that were contacted. c is the minimum number of contributors in the project whoe were invited to participate and r is the number of respondents who answered the questionnaire

Project	**Description**	**c**	**Cont.**	**r**	**Respondent Countries**
Mplayer	A multimedia player for Linux	16	yes	1	Germany
The GIMP	An image editing program	6	yes	5	USA, Germany, Australia, France
Kopete	An instant messaging client	6	yes	5	Brazil, Chile, Germany, UK, Turkey
GNU cash	A budget program for Linux	1	yes	1	UK
Gnome	Desktop environment	50	yes	1	Denmark
Abiword	An open source word processor	11	yes	1	UK
mmsv2	A system to play multimedia files on a TV	1	yes	2	Denmark, Sweden
Gnumeric	A spreadsheet application for Gnome	1	yes	2	Denmark, Australia
Xine	A multimedia player	11	yes	1	Germany
Inkscape	A vector drawing program	16	yes	0	
Konqueror	KDE's file manager and browser	33	yes	0	
Kile	A LATEXsource editor	29	yes	0	
K3B	A CD-ROM / DVD burning application	4	yes	0	
Amarok	A multimedia jukebox	5	yes	0	
Point of sale	Proprietary retail inventory software	1	no	1	Belgium
Tuxpaint	A children's painting program	50	no	1	USA
Tapper	A software installation manager for Linux	1	no	1	Norway
Scribus	A Desktop Publishing application	50	no	1	Canada
Kshower	A visualization tool for data	1	no	1	Greece
Total		**293**		**24**	

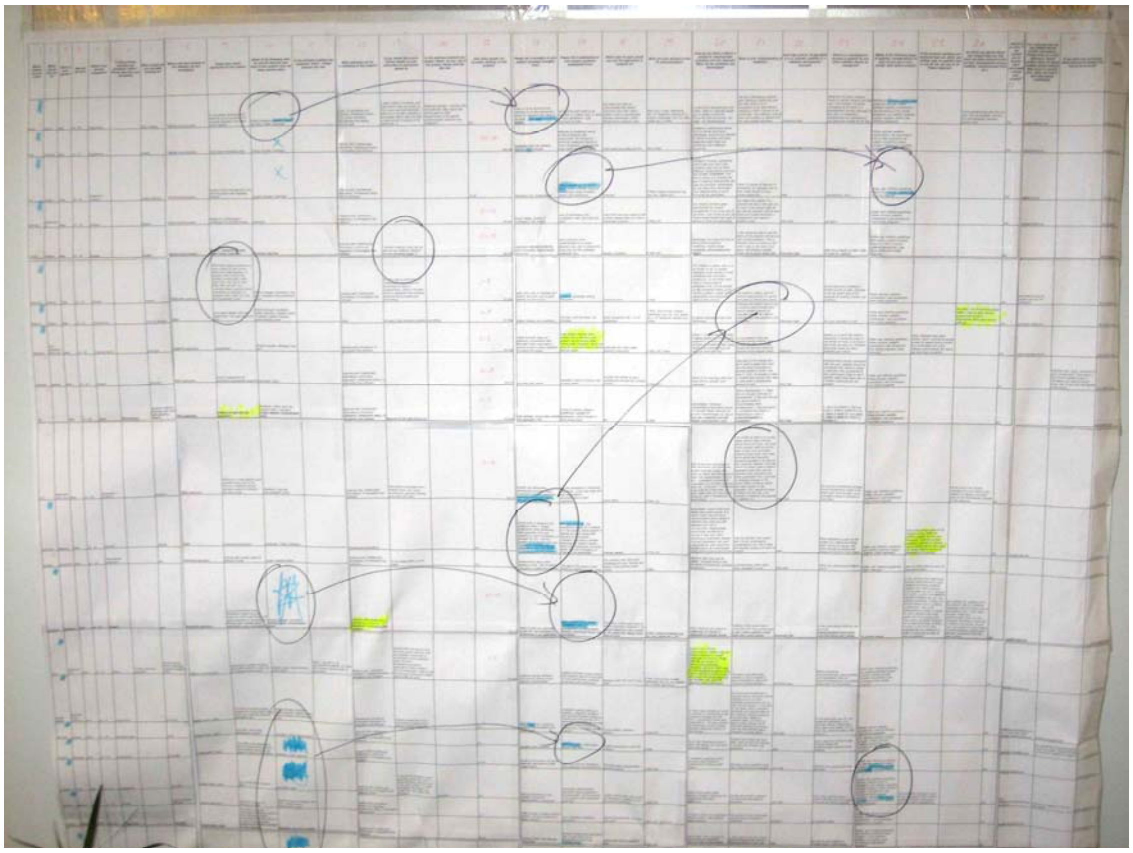

Figure 1. A mapping of the questionnaire results

3.2. Developer Interviews

To get a clear picture of the methods and thoughts connected to usability evaluation in OSS projects, we chose to perform interviews with three of the respondents from the questionnaire survey. In the questionnaire, we asked the respondents if they were willing to help us further by participating in a personal interview, and eighteen out of the twenty-four respondents answered that we could contact them for more information about their project.

Participants: Since the main purpose of the interviews was to acquire knowledge about usability within open source software, we looked for respondents who had the role of usability tester or project manager. The role of project manager was relevant since we wanted to attain knowledge on how OSS projects were administered in regards to usability. In the questionnaire, respondents indicated the various roles that they had in their OSS project. We used this to select respondents for follow-up interviews. This screening process resulted in the selection of three OSS contributors for the follow-up interviews, two project managers and one usability tester. One of the project managers and the usability tester were involved in the Kopete project, and the second project manager was the main developer on mmsv2.

Procedure: The interviews were conducted as semi-structured interviews, based on an interview guide to make the results comparable, and the interview guide was based on a set of thematic questions [0]. We investigated the following themes:

- Respondent's motivation for contributing to OSS
- Usability considerations used in the project
- Frequency and place of usability evaluations
- Usability as a part of the development process
- Usability experts in the development team
- Willingness to alter program code because of usability problems discovered in tests
- Decision making in the project, especially regarding usability

Data Collection: As the respondents were located in three different countries we contacted them separately and suggested interview methods according to what was possible and convenient for them. This resulted in one direct interview and two interviews via instant messaging software. In the direct interview, we used an audio recorder. Afterwards, the recording was transcribed into 9 pages of text. The interviews conducted through instant messaging were transcribed through a built-in feature, which saved the conversation in an XML file.

Data Analysis: In the data analysis of the interview we also used meaning condensation. First we identified a number of topics or tendencies we found important in the transcriptions. Following this, we analyzed the statements in more detail to extract the overall opinion of the respondent.

3.3. Evaluator Interviews

To get a complementary perspective on how usability tests were conducted in OSS projects, we contacted the company Relevantive in Berlin. Relevantive is currently a leader within usability in OSS projects, and they develop and administer the website openusability.org, where they provide communication between OSS projects and usability evaluators.

Participants: Five employees at Relevantive participated in this activity. They were all experienced in OSS development and usability engineering.

Procedure: We performed the interviews over six hours. First, we conducted a two-hour focus group interview with all the five employees. This was conducted as a semi structured interview, based on an interview guide with the following seven themes:

- Background information about Relevantive
- Test procedures of OSS
- Usability evaluation
- Communication with OSS developers
- K Desktop Environment (KDE) guidelines
- Remote usability evaluation
- OSS and usability in general

Second, we had personal interviews with two of the employees. For these interviews we did not use a pre-constructed interview guide, but instead we used the data collected from the focus group interview to find themes to explore further. Third, we had the opportunity to observe them while they conducted a usability test of an OSS product.

Data Collection: We used a combination of audio recording and note taking to collect data. The focus group interview was recorded on a laptop, and later transcribed into 15 pages of text. The individual interviews were not recorded, but three moderators took notes. Immediately after the individual interviews, the three moderators compared notes and compiled these in a single document with 6 pages of text.

Data Analysis: For the data analysis we used the same procedure as in the developer interviews.

4. Results

Through the empirical studies we attained knowledge on the perception of usability in OSS by both developers and usability professionals. Furthermore, we got an insight in the current practice of usability evaluation in OSS projects. The results will be presented in terms of the following four themes:

- The OSS contributors
- Opinions about usability
- The OSS development process
- Usability evaluation methods in practice

4.1. The OSS Contributors

OSS contributors are young males. In the questionnaire, the respondents primarily consisted of males between 19 and 40. This is comparable to the Boston Consulting Group survey that found the average age of OSS contributors to be 30 years [0].

Contributors to OSS are highly ideological. Initially we wanted to investigate who the OSS contributors were and what their motivation for participating in OSS development was. In the questionnaire and in the interviews, we found that the main motivation for contributing to OSS for both developers and usability experts was ideology. In the questionnaire answers, 88% of the developers chose 'To strengthen free software' as their motivation, see Figure 2, and the interviews with developers and usability professionals supported this. In addition, 54% of the questionnaire respondents chose 'Community reputation' as a motivation.

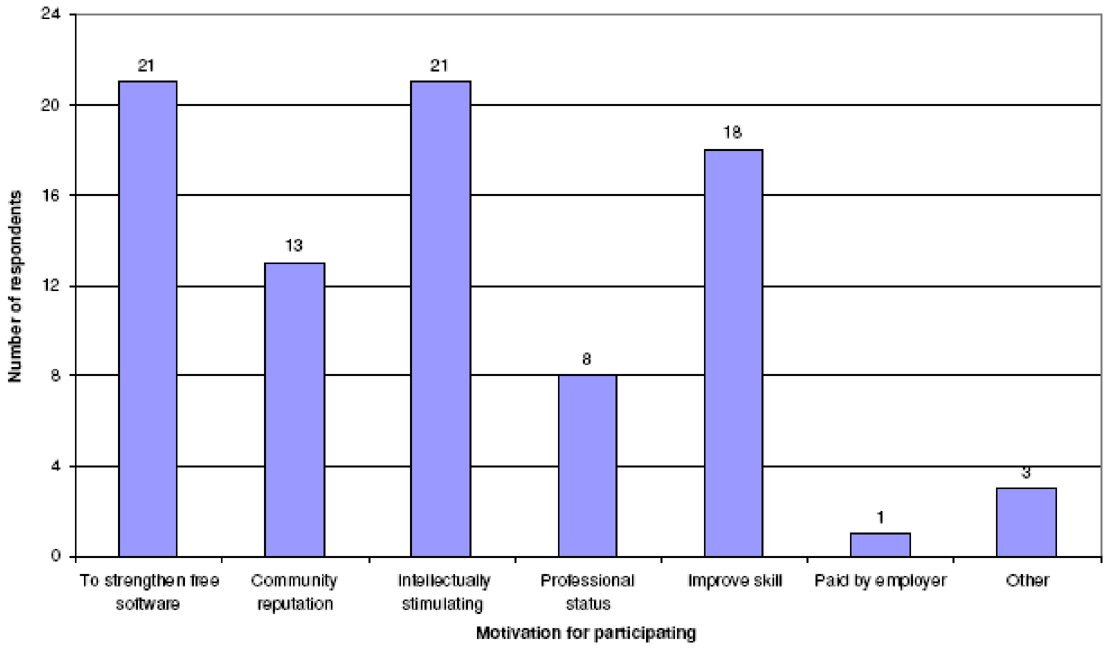

Figure 2. The motivation for developers to contribute to OSS. The 24 respondents could choose more than one motivating factor

Contributors to OSS want challenges. 75% of the developers contribute to OSS in order to improve their skills and 88% wanted to be intellectually stimulated. The technical challenge as a motivating factor was also evident in the interviews with OSS contributors. A project manager explained that the idea for his OSS project came about "because at the time when I started it, there weren't any multimedia systems actually that was either open source or closed software". Another project manager explained that he initially needed a program to chat with his girlfriend and therefore wanted to develop a system that allowed him to do so. Furthermore, one third of the questionnaire respondents stated that they were motivated to contribute to OSS because of 'Professional status'. We interpreted this to be similar to improvement of skill in the way that OSS contributors feel that they develop themselves and gain knowledge through their involvement in OSS. The interviewed usability professionals explained that they got valuable experience from their involvement with OSS. The immediate effect of a usability evaluation was very satisfying and a great motivation for further involvement in OSS. For instance the results of Relevantive's usability evaluation of the German edition of Wikipedia, an open source on-line encyclopedia, were also implemented in the international edition of Wikipedia. This differed greatly from their co-operation with commercial software companies, where consultant reports often resulted in only minor changes to the software, or none at all.

4.2. Opinions about Usability

OSS contributors consider usability as important. 83% of the questionnaire respondents regarded the importance of usability as either 'high', 'very high' or 'extremely high'. Only 13% considered it 'moderate', 4% stated 'slight' and nobody thought it had no importance, see Figure 3. Two of the OSS developers we interviewed had stated in the questionnaire that usability had extremely high importance. One of them explained that some developers see usability as a trivial task which is neither interesting nor intellectually stimulating; "...hackers code for fun, and sure it is more fun to add support for some protocol feature than fixing a dialog for grandma". The third developer we interviewed stated in the questionnaire that usability had only slight importance. In the interview he explained that he developed the system as a hobby and that a large user base was not his goal. The usability experts at Relevantive considered the importance of usability as being extremely high. Their experience was that especially the young OSS developers had usability as a high priority while some of the older developers were reluctant. We did not have enough information about the respondents in the questionnaire to support this notion though.

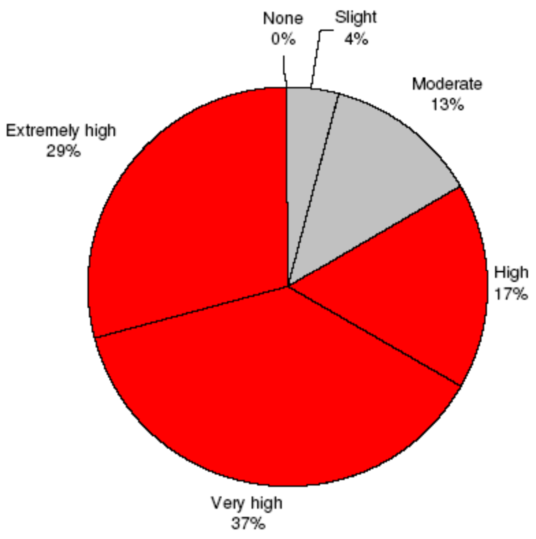

Figure 3. The priority of usability

There is considerable divergence in the definition of usability. The way the OSS contributors defined the term 'usability' in the questionnaire varied considerably. We divided their definitions into three aspects corresponding to the ISO-9241 definition: effectiveness, efficiency and satisfaction [0]. Definitions that did not relate to any of these aspects were categorized as either 'Technical property' or 'No category', see Table 2.

Table 2. The 24 definitions of usability distributed on aspects. The total of 28 occurs because 4 definitions cover two aspects

Aspect	**Keywords**	**n**
Effectiveness	Accuracy, completeness, productivity	7
Efficiency	Learning time, intuitiveness, resources spent	16
Satisfaction	Attitude to system, entertainment value	1
Technical property	Functionality of system	3
No category	Did not provide definition	1
Total		**28**

Efficiency was the key aspect in 16 of the definitions. One respondent defined it this way: "Usability is the science that's concerned with how quickly/easily a user is able to perform useful tasks with a given system". Some of these definitions were focused on intuitiveness and put emphasis on the affordance of the user interface and user-centered design: "A user should be able to use the basics of the program without any help of documentation. This is basically done by building a GUI, which maps to the user's mind space; like putting a dustbin where you're gonna delete files". Another respondent defined usability very brief but still emphasized the importance of intuitiveness "If your grandma can use it".

Seven definitions focused on effectiveness. For instance one respondent stated that usability was: "Allowing a user to perform tasks with as little diversion as possible". Three respondents defined usability based on some technical attribute of the program. One of them defined usability "Within development: 1. Tools that do not get in the way of development. 2. Security through the use of GnuPG. After development: 1. Documentation. 2. Accessible bug reports. 3. Responding to users. 4. Internationalization. 5. Localization". Another simply defined usability as "A working thing, that works when requested to work". These definitions revealed an alternative, developer-centered understanding of the term usability. These definitions show that a high number of the OSS contributors who participated in the questionnaire survey understood usability in a very simplified way and that there was a lot of divergence between them. Only four definitions covered more than one aspect, and none of them covered all three aspects. This divergence shows that although there appears to be agreement about the high priority of usability, there is not agreement about the meaning of this term.

Usability experts are only advisors. Although most OSS contributors wanted a higher degree of usability in their software, they were reluctant to include usability experts directly in the development process. OSS contributors clearly stated that they were afraid that direct involvement of a usability expert, especially in decision making, would overrule the democratic nature of OSS development since it would be difficult to have a democratic debate against the only expert on the matter; "Makes no sense to have one person deciding how the interface should look. I prefer the independent group approach. Fits better in the OSS model". Another contributor described the possibility of contributions from usability experts as: "It's a bit difficult. OSS people don't like too much to be told what to program. The developers put in resources in accordance with their personal interests, and maybe a usability expert by itself would not be sooooo useful". However, OSS developers were positive towards external usability evaluations where experts produced a usability report of the tested program: "From time to time we get some usability reports from professional people. ... Once in a while they arrive and bless us with their wisdom". Despite the appreciation of the knowledge of usability professionals and the usability reports, these expressions conveyed a gap between the technically minded contributors and those with a usability background.

4.3. The OSS Development Process

The OSS development process is characterized by short iterative cycles. All developers expressed that the development process is characterized by small iterative steps. Two of the developers compared OSS development to the extreme programming (XP) development paradigm [0]; "I'm an XP fan, so I start doing things in short steps. I add functionality and elements to the interface as needed, but try to group them in a meaningful way". This iterative approach enables inclusion of usability evaluation in each cycle, but that does not occur in practice. One of the interviewed OSS contributors stated about the recommendations of an external usability evaluation that was more than six months old: "I have to confess most of this stuff is not yet implemented". Therefore the short iterative cycles of the OSS development process need to be considered by usability professionals providing feedback to developers; feedback needs to be realistic and possible to implement within the current release cycle.

Usability is regarded as an add-on property. A typical understanding of usability was that it could be incorporated at a certain stage of the development process, for instance once the program could be compiled or had the desired functionality. The analysis showed that the developers had different ideas on when usability belonged in the development process. We identified four main stages (one respondent was not sure what to answer):

- In the beginning (12)
- Iteratively (5)
- In the end (1)
- During testing / QA (5)

This showed that there was no shared opinion about where in the development process the main usability effort should be made. Still, half of the respondents stated that it should be considered in the beginning of the development process. For instance one respondent stated that the usability effort should be "At the beginning. Usability is harder to bolt on later, although it can be added later at the expense of creating a whole new interface". There was a clear difference of opinion even between developers contributing to the same OSS project. For instance the contributors to the Kopete project assigned usability considerations to four different stages. The usability professionals at Relevantive shared the impression that OSS developers often saw usability as an add-on property of the software and not as an integrated part of the development process.

Democracy is vital in the OSS development process. Compared to traditional software development there is little formal leadership. We noticed that even though almost every OSS project had at least one project manager associated, this title did not imply leadership overthe project. Often this title reflected the person who founded the project rather than the person who kept track of everything or delegated tasks to other contributors. Still, statements were ambiguous "I am the original author of Kopete. Kopete has no project manager. I am still the benevolent dictator. We have hardcore contributors, release dudes, etc. but nobody manages the project". Another of the interviewed persons stated "Some persons, the 'fathers' of the project, have an outspoken voice and can persuade more easily about some issues" The project managers we interviewed stated that one of the main concepts of OSS was the democratic way of developing software. Most major decisions were made democratically by everyone involved and discussed for instance on the mailing list.

Trust is crucial in the development process. Cooperation within the OSS community is based on trust and both developers and usability professionals contributing to OSS need to be prepared to build a rapport with other contributors. For instance Relevantive experienced that almost all problems faced when working with OSS developers were grounded in lack of trust, which made developers ignore suggestions from usability professionals. On the other hand the developers stated that a change suggested by usability evaluators "has to make sense, and we need to evaluate if it can be done. Some changes are too big to be done, not because of the idea itself, but because of the underlying code". The nature of OSS, where contributors rarely meet in person, makes it necessary to judge others based on past merits. It can be difficult for a usability expert to display merits, since usability improvements are more difficult to measure than the programming of a new feature. Relevantive stated that when there was no face-to-face contact with the other person, building trust could be the most strenuous task of co-operating with OSS developers. Relevantive found that attending various OSS conferences and gatherings was an excellent way to get acquainted with OSS developers and ultimately build the necessary level of trust. When trust had been established, the direct contact between usability professionals and developers resulted in a work environment, which, in the view of Relevantive, was very gratifying.

4.4. Usability Evaluation Methods in Practice

Common sense is the primary evaluation method. In the questionnaire survey 79% of respondents answered that they followed common usability conventions and the same number of respondents stated that they used usability guidelines. Active usability evaluation was not used as frequently; 42% answered that they used expert inspections, but they were rarely performed by usability professionals, 21% mentioned a remote usability evaluation and about 8% used a usability laboratory, see Figure 4.

Guidelines replace usability evaluation. The methods most often used were formal or informal guidelines, and inspirations from similar programs. One set of guidelines often mentioned in the interviews was the KDE user interface guidelines. These guidelines define standards for menu layouts and user interface structure within the KDE and in addition it also provides a programming framework for application design. In the interviews the framework was highlighted as a usability-enhancing factor since "basically the only way to escape the guidelines is when you try to make a strange IU component, which is not part of the framework". Nevertheless, the interviewed usability professionals rejected the notion that this would ensure a high degree of usability, even though some of them had been deeply involved with the development of the KDE user interface guidelines. Although they considered standards and consistency to be some of the most important parts of usability, they firmly stated that guidelines alone were not sufficient; However, the usability professionals also said that "guidelines can only be made for general items and not for specialized functions" and they did not think guidelines could replace usability evaluation.

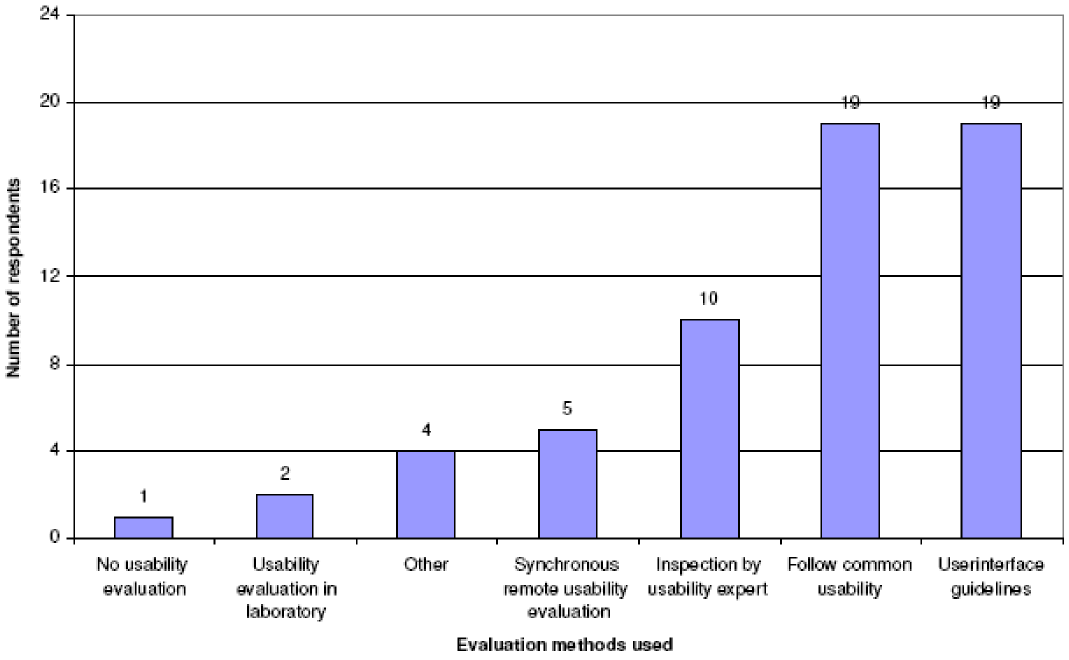

Figure 4. The developers were asked which evaluation methods were used in their project if any. The respondents could choose more than one method

Money is a deciding factor. Only two of the respondents mentioned that they had used laboratories for usability evaluation. One of the interviewed developers thought that the economy of using a laboratory was a key issue: "I think that the usability aspect is sort of harder for open source projects to do, so some sort of easier way or cheaper way to do this would actually be very welcoming, I would think, because you can't rent a decent lab". Usability evaluations of OSS conducted by Relevantive were often performed as inspections by the professionals themselves. This testing method was not chosen because it was deemed the best, but often it was the only possible test form. In most cases, Relevantive did not receive any funding to finance their evaluation of OSS software. Hence, they were not able to pay 'real users' to participate in conventional usability testing in a dedicated laboratory, even though they considered this a more effective evaluation method.

Remote usability evaluation is not a substitute for a laboratory evaluation. Five questionnaire respondents indicated that they used synchronous remote usability evaluation methods, see Figure 4. When we investigated this further during the interviews, we discovered that their idea of remote usability evaluation had nothing to do with usability evaluation. For example, one described it as connecting to a users computer and performing 'live' bug-fixing. The usability professionals at Relevantive were hesitant towards using remote methods where a conventional think-aloud evaluation was performed remotely. In their opinion facial expressions were vital for identifying usability problems. A remote setup would also introduce too many problems, if the test users had to setup an environment consisting of web-cams, shared desktop and audio connections by themselves. A number of studies of remote usability testing mentioned the benefits of letting the test user operate in their normal working environment without the stress of a room full of recording equipment and observers [0, 0, 0, 0, 0].

5. Discussion

During our empirical study we have noticed the following three common mantras about OSS and usability that we will discuss in this section:

- OSS development is always democratic
- OSS will solve the 'software crisis'
- Usability problems are just bugs

5.1. OSS Development is Always Democratic

We found that OSS projects in general had flat organizational structures and that OSS developers in our study praised the democratic organization of the development process. Furthermore, 25% of the respondents in the questionnaire survey indicated that they contributed because of ideological reasons. The survey by Boston Consulting Group showed similar results [0].

It can, however, be questioned whether the OSS development process is indeed democratic. Admittedly the development process and the flow of communication is open to anyone, but the typical situation with no formal leader is not necessarily a sign of democracy. Raymond instead chose to describe the OSS development model as 'Meritocratic', indicating that the previous contributions of the developers founded the basis of the social structure and influence on future decisions [0]. Trudelle stated that UI designers involved with OSS should be "willing and able to engage the beast, for they can only get the needed leverage from impressing those doing the work - it's a meritocracy out there" [0].

Our interviews with both Relevantive and the OSS developers revealed the same mechanism; a level of trust must be built through merits in order to gain influence on the development process. One of the interviewed OSS contributors jokingly referred to himself as "the benevolent dictator" because he had initiated the project. Others confirmed that there is in fact a level of hierarchy among OSS contributors. However, these sociological patterns of OSS still need further research to be fully understood.

5.2. OSS Will Solve the 'Software Crisis'

The notion of a 'software crisis' has survived during the last 30 years of computing. It refers to the problems of delivering quality software that fulfill the requirements of the users, and staying within budgets and deadlines [0]. It has been argued that no single development will solve this crisis. In contrast, it has been claimed that OSS development could possibly solve some of these problems [0]. Arguably, the OSS approach is not a full-scale development model but rather an underlying philosophy with a set of principles to use during development. In our interviews, the OSS contributors compared OSS development to XP in regards to small development cycles and a well-maintained prototype. Methods like XP have evolved as a reaction to shortcomings of conventional software development when it comes to rapid software development [0]. The bazaar approach of OSS [0] combined with the emphasis on testing in XP has at least the potential to solve some of the problems in regard to delays and code quality.

5.3. Usability Problems are just Bugs

It has been suggested that there are two fundamentally different approaches to track usability issues. One is to handle usability issues as regular bugs that are added to a bug database like any other problem. The other is to have a separate infrastructure to handle usability issues [0]. The usability professionals at Relevantive have taken the latter approach. The disadvantage of separating usability issues from other bugs in the system is that it puts usability out of the mainstream of development and makes it less visible to the developers. Furthermore, if usability issues are not in the general bug reports, they are less likely to be fixed. On the other hand comparing usability issues to programming bugs is a risk, since a bug that makes the system crash obviously will get a high priority compared to a 'cosmetic' problem in the user interface. In addition, some bug databases do not have sufficient categories to fully describe usability issues [0].

In our study we found that OSS developers prioritized an open and free negotiation process of potential changes to the project. Such a negotiation process within OSS has been studied through examination of negotiations in the context of the OSS bug report database Bugzilla. It was found that negotiation takes place in 61% of bug reports and 27% contains evidence of negotiation of several issues [0]. The open negotiation process can sometimes compromise decision making. Netscape's experience in the Mozilla project was that although the open discussion where everyone could participate was beneficial, sometimes it was more productive to create a 'by-invitation' group of UI owners and stakeholders to settle issues and make authoritative decisions [0]. These studies illustrate the progress of current usability initiatives. It is still not clear whether it works better than simply treating a usability problem as a bug.

6. Conclusion

In this study we explored opinions about usability among developers and usability experts involved with OSS. Furthermore, we gathered information about the current practice of usability evaluation. Overall we found that OSS developers are interested in usability, but in practice most of the efforts are based on common sense. They appreciate external usability evaluations performed by professionals, as long as these professionals are not interfering in decision-making about changes and priorities.

Usability evaluation is not the top priority of many OSS projects. OSS contributors are mainly motivated by an ideological belief in free software and to be intellectually stimulated. Yet awareness of the topic is increasing among OSS developers. Currently, most developers have a very limited understanding of usability. Moreover, there is a lack of resources and evaluation methods fitting into the OSS paradigm.

The results presented in this paper are based on an empirical study with three elements. Even though we made a considerable effort to get response from many OSS projects, the total number is limited. The follow-up interviews were also limited in numbers. The projects we have studied are mostly of the classical kind with non-employed volunteers. Recently, large software organizations have moved into OSS development, and such projects may give other conclusions.

The study shows that the current practice in OSS rarely employs systematic usability evaluation and when it does, it is well-known methods from standard usability evaluation. It would be interesting to experiment with other methods that fit better to the nature of OSS development processes.

Acknowledgments

We would like to thank all the participants in the questionnaire survey, and Anders Rune Jensen, 'Taupter' and Duncan who participated in the interviews. Furthermore, we would like to thank Ellen Reitmayr, Björn Balazs and the rest of the staff at Relevantive for their co-operation and hospitality during our visit. Finally, we also want to thank the anonymous reviewers.

References

[1] **V. Bartek, D. Cheatham.** Experiences in Remote Usability Evaluations. 2004. *http://www-3.ibm.com/ibm/easy/eouext.nsf/Publish/50?OpenDocument&../Publish/1116/$File/paper1116.pdf.*

[2] **K. Beck.** Extreme Programming Explained: Embrace Change. *Addison-Wesley*, 2001.

[3] **C. Benson, M. Muller-Prove, J. Mzourek.** Professional Usability in Open Source Projects: Gnome, openoffice.org, netbeans. *CHI'*04 *extended abstracts, ACM Press*. 2004, 1083-1084.

[4] **J.F.P. Brooks.** No Silver Bullet: Essence and Accidents of Software Engineering. *Computer*, 20(4), 1987, 10-19.

[5] **S. Dray, D. Siegel.** Remote Possibilities? International Usability Testing at a Distance. *Interactions*, 11(2), 2004, 10-17.

[6] **S. Eklund, M. Feldman, M. Trombley, R. Sinha.** Improving the Usability of Open Source Software. *Usability Testing of staroffice calc.*, 2001, *http://www.sims.berkeley.edu/~sinha/opensource.html.*

[7] **J. Feller, B. Fitzgerald.** A Framework Analysis of the Open Source Software Development Paradigm. *Proceedings of ICIS, ACM*, 2000, 58-69.

[8] **N. Frishberg, A.M. Dirks, C. Benson, S. Nickell, S. Smith.** Getting to Know You: Open Source Development Meets Usability. *CHI* '02 *extended abstracts, ACM Press*, 2002, 932-933.

[9] **E. Frøkjær, M. Hertzum, K. Hornbæk.** Measuring Usability: Are Effectiveness, Efficiency and Satisfaction Really Correlated? *Proceedings of CHI'*00, ACM Press, 2000, 345-352.

[10] **L. Gasser, G. Ripoche.** Distributed Collective Practices and f/oss Problem Management. *Proceedings of the Conference on Cooperation, Innovation and Technologie, CITE*2003, 2003.

[11] **D. Gough, H. Phillips.** Remote Online Usability Testing: Why, How, and When to use it, 2003. *http://www.boxesandarrows.com/view/remote.*

[12] **J. Johnson-Eilola.** Open Source Basics: Definitions, Models, and Questions. *Proceedings of the 20th annual international conference on Computer documentation, SIGDOC'*02, *ACM Press*, 2002, 79–83.

[13] **S. Kvale.** Interview – En introduktion til det kvalitative forskningsinterview. *Hans Reitzels Forlag*, 2001.

[14] **K. Lakhani, B. Wolf.** BCG Hacker Survey. Boston Consulting Group, 2002. *http://www.ostg.com/bcg/BCGHACKERSURVEY-0.73.pdf.*

[15] **D.M. Nichols, M.B. Twidale.** Usability and Open Source Software. *Technical Report* 10/02, *Department of Computer Science, University of Waikato, Working Paper Series*, 2002, ISSN 1170-487X.

[16] **P. Nielsen.** Produktion af viden – en praktisk metodebog. *Teknisk forlag*, 1998.

[17] **S. Pemberton.** Scratching Someone Else's Itch: Why Open Source Can't Do Usability. *Interactions*, 11(1), 2004, 72.

[18] **E.Raymond.** The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. *O'Reilly*, 1999.

[19] **E.Raymond.** The Revenge of the Hackers. *O'Reilly*, 1999.

[20] **E.Raymond.** Why Open Source Will Rule, 2002. *http://news.zdnet.com/2100-3513 22-871366.html.*

[21] **M. Safire.** Remote Moderated Usability, 2004. *http://www.upassoc.org/usability.resources/conference/2004/im safire.html.*

[22] **R.J. Sandusky, N.L. Gasser.** Egotiation and the Coordination of Information and Activity in Distributed Software Problem Management. *Proceedings of the* 2005 *international ACM SIGGROUP conference on Supporting group work, GROUP'*05, *ACM Press*, 187-196.

[23] **R.J. Sandusky, N.L. Gasser, G. Ripoche.** Bug Report Networks: Varieties, Strategies, and Impacts in a f/oss. Development Community. *International Workshop on Mining Software Repositories, MSR* 2004. *IEEE International Conference on Software Engineering.*

[24] **Smith, Engen, Mankoski, Frishberg, Pedersen, Benson**. Gnome Usability Study Report. 2001. *http://developer.gnome.org/projects/gup/ut1report/reportmain.html.*

[25] **R. Stallman.** gnu General Public License, 1991. *http://www.gnu.org/copyleft/gpl.html.*

[26] **K.E. Thompson, E.P. Rozanski, A.R. Haake.** () Here, There, Anywhere: Remote Usability Testing that Works. *Proceedings of the 5th conference on Information technology education, CITC*5'04, *ACM Press*, 2004, 132-137.

[27] **P. Trudelle.** Shall We Dance? Ten Lessons Learned From Netscape's Flirtation with Open Source UI Development. *The Open Source Meets Usability Workshop, Conference on Human Factors in Computer Systems,* 2002.

[28] **M.B. Twidale, D.M. Nichols.** Exploring Usability Discussions in Open Source Development. *Proceedings of HICSS'*05, 2005.

[29] **C. Wilson, K.P. Coyne.** The Whiteboard: Tracking Usability Issues: To Bug or not to Bug? *Interactions*, 8(3), 2001, 15-19.

Received August 2006.