Integrating HCI Specialists into Open Source Software Development Projects

Henrik Hedberg and Netta Iivari
University of Oulu, Department of Information Processing Science
P.O.Box 3000, FI-90014 University of Oulu, Finland
{henrik.hedberg, netta.iivari}@oulu.fi,
WWW home page: http://www.tol.oulu.fi/o2s2/

Abstract. Typical open source software (OSS) development projects are organized around technically talented developers, whose communication is based on technical aspects and source code. Decision-making power is gained through proven competence and activity in the project, and non-technical end-user opinions are too many times neglected. In addition, also human-computer interaction (HCI) specialists have encountered difficulties in trying to participate in OSS projects, because there seems to be no clear authority and responsibility for them. In this paper, based on HCI and OSS literature, we introduce an extended OSS development project organization model that adds a new level of communication and roles for attending human aspects of software. The proposed model makes the existence of HCI specialists visible in the projects, and promotes interaction between developers and the HCI specialists in the course of a project.

1 Introduction

Open source software (OSS) development projects produce more and more applications for a large user population, which does not consist only of OSS developers themselves anymore. In addition, during recent years firms have started to consider how to gain competitive advantage from OSS. Software (SW) companies are utilizing OSS as part of their products or releasing the source code of their products and participating in OSS communities developing it further. OSS is, therefore, no longer developed only to serve particular developers and their needs. [15, 31] Instead, there will be more and more users without deep technical knowledge. That leads to higher quality expectations than earlier, when an end-user was seen as a co-developer tolerating crashes, hunting defects, and fixing the code.

Usability is an important quality characteristic of software products and systems. Especially the field of Human Computer Interaction (HCI) has addressed the development of usable software products and systems, which has resulted in the development of approaches such as usability engineering (UE) and user-centered design (UCD) (e.g. [18, 22, 23, 26, 32, 34]). It is generally admitted that many OSS

projects neglect usability, instead producing user interfaces by engineers to engineers and thus providing weak usability [1, 29, 30, 36, 39].

In this paper we acknowledge that the end user population - including the 'naive', non-technical, non-computer professional users - is constantly growing, and therefore usability of the OSS should be improved [1, 3, 14, 29, 30, 33, 39, 40]. We bring out the limited influence of non-technical end-users and the non-existence of HCI specialists in traditional OSS development projects. We also search the root cause from the OSS projects' decision-making structure that underlines technical competence, thus being a very challenging environment for HCI specialists to co-operate with developers.

As a result we propose an extended OSS development model that adds a new level of communication and roles for human aspects of the software. We introduce a decision-making structure that defines a clear mechanism for integrating HCI specialists into OSS projects that are willing to provide better user experience. The model is grounded on the knowledge of the existing OSS project structures and the HCI literature as well as on our own experiences in real OSS projects. We also describe the initial evaluations made and the ongoing experiment, which are used to adjust the proposed model.

This paper is structured as follows: Section 2 covers the typical relationship between usability and OSS projects, and section 3 describes the OSS project organization model based on the variations suggested by several authors. The background for the proposed solution is compiled from HCI literature in section 4, and the the actual model is proposed in section 5. Finally, section 6 contains the initial evaluation of the proposed model, and section 7 presents conclusions.

2 Usability in OSS Projects

There are some examples of high quality OSS, but in general OSS usability tends to be weak [1, 29, 30, 36] because user interfaces are produced by engineers for engineers [3]. It is largely due to historical background of OSS: developers are writing software for themselves. Under those circumstances, there is no need to cater for non-technical end-users. [5, 33] That lead to overrated expectations of users' knowledge about the operating system, the invisible technical details, and the language the developers use [28, 29]. Although user involvement has been argued to be the essential characteristic of the OSS success, it has become one major problem from the usability point of view: it should be acknowledged that usually the developer-users alone do not and can not represent the non-technical users [28, 39, 40].

Also in OSS projects intended for larger audience, the user profiles might be unclear or nonexistent [3, 5, 6]. Non-technical users are naturally not able to debug or fix bugs, and also typical communication channels in OSS projects, i.e. mailing lists and bug repositories, may be too technical [3, 6, 29] and not suitable for usability issues [6, 29, 30, 36]. Thus, the number of reported user interface problems

is low in many OSS projects [39]. In addition, typical OSS projects with no HCI specialists do not gather feedback from the non-technical end-users [3, 5, 6, 29, 30, 37].

Due to the lack of usability in many OSS projects, the literature suggests participation of HCI specialists in the OSS development (e.g. [1, 3, 6, 29, 36, 39]). It is acknowledged that just getting the end users involved is not enough because they are not trained for developing and ensuring usability, even though they encounter the actual usability problems while using the system [39, 40]. However, it is argued that decentralized and engineering driven OSS development is in contrast with corporate processes which also UCD and UE methodologies represent. Those are felt too heavy-weight processes and being against "open source philosophy". [3, 5, 6, 39] Another conflicting issue to be considered by the HCI community is the need for distributed design in the OSS development context. In HCI, support for this has not been considered much. [6, 29, 30]

On the other hand, the participation of companies in OSS development has been seen as an enabler for usability activities, since the companies can provide both professional usability resources and HCI guidelines [1, 3, 6, 29, 37]. Although it has to be acknowledged that there are a variety of OSS projects, many OSS development efforts would benefit from the active co-operation with HCI specialists. The interesting challenge is how to combine usability efforts with the OSS development model.

3 Decision-Making in OSS Projects

The OSS project organization is often depicted as a hierarchical or onion-like structure [10, 9, 17]. Usually the decision-making power is centralized on one developer, like in Linux kernel development, or on several developers forming a core team, like in Apache HTTP Server project. Because of their technical capabilities and activity, the core team is being respected and having authority to make the decisions related to what to include in the code base (e.g. [13, 31, 38]).

The next in the hierarchy are other active participants, often called as co-developers [10, 17]. However, it must be noticed that, regarding to decision-making, there are two kinds of developers: those who have the right to modify directly the source code in the version control system, and those who send patches to project hoping that those are applied into the actual release. Following the naming policy of the Apache HTTP Server project [2] and the FreeBSD project [16], we call the first group as committers, since they have the right to commit, or write, to the source code repository, and the latter as contributors.

The outer layers of the OSS project structure consist of active and passive users. Active users are able and willing to participate by writing, for example, bug reports, feature requests or documentation, but not the actual source code. Passive users are just using the software without affecting the course of the project. [10, 17] Based on

this analysis, our view of typical OSS development project structure is depicted in Figure 1.

The decision-making power is tightly in the hands of developers in OSS projects. The position in the hierarchy depends on the previous contributions, and almost only way to gain more authority is to participate in the development in the source code level. Due to this meritocracy, it might be difficult for HCI specialists with lack of technical expertise to influence the course of the project. The OSS developers may welcome HCI specialists as advisors, but they do not want to give them decision making power regarding the solution. [1] Furthermore, in OSS development developer criticism/feedback tends to be considered much more important than end-user criticism/feedback [24]. The most preferred way to communicate solutions and alternatives is to implement those first, and evaluate those afterwards. Unfortunately that is not the ideal way to work with HCI issues or for non-technical HCI specialists and users. It is clear that the existing OSS development models do not recognize the existence of HCI specialists in OSS projects.

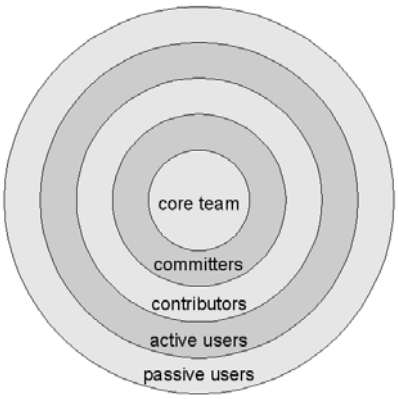

Fig. 1. The typical OSS development project structure with roles of different stakeholder groups

It has to be pointed out that also IT companies are getting involved with OSS development. Thus, it needs to be acknowledged that the companies may want to influence the community and be able to take part in decision-making. It has been argued that companies are the ones that need to adapt [19] and they should try to adhere to the community values and spirit to be accepted by the community [15]. On the other hand, researchers have also already shown that the companies and the OSS communities can end up with many different kinds of relationships, and in some cases the company clearly can influence the community [11]. This opens interesting possibilities for integrating HCI specialists into OSS projects.

4 HCI Literature

A number of UCD and UE methodologies have been developed, their principal goal being making systems usable. Also the principles of active user participation, appropriate allocation of functions between users and technology, iterative design and multi-disciplinary design have been associated with these approaches. (e.g. [18, 21, 22]). Despite their differences (see e.g. [21]), all these methodologies emphasize the importance of gaining a detailed understanding the user, his/her tasks or work practices and the context of use as a basis. It is emphasized that it is essential to understand and specify who are the intended users, what are their goals and tasks in using the system and in what kind of environment (social, technical, cultural etc.) the system will be used (see e.g. [4, 8, 26, 32, 34]).

Afterwards, one should carefully redesign the tasks or work practices based on the understanding. It is maintained that interactive systems necessarily define new ways to work. These new work practices might be only implicitly designed, but anyhow delivered through the solution. However, an explicit redesign of users' work and tasks should be carried out. (See e.g. [4, 8, 26, 32, 34].) After user task redesign, human computer interaction solutions are to be produced, following the state-of-the-art HCI guidelines. Related to the design solutions, furthermore, user feedback should be sought early and the solutions iterated based on the user feedback. The evaluation should be started early in order that the results affect the design and making changes is not expensive. The emphasis is on gathering qualitative feedback. Later during the development one should measure whether the requirements have been met. [22, 35]

As mentioned, user participation is also an integral element of UCD. However, very different approaches to it can be adopted. Damodaran [12] has classified users' role as informative, consultative or participative. In informative role users act only as providers of information and as objects of observation (e.g. during field studies aiming at understanding the users and the context of use), in consultative role they are allowed to comment on predefined design solutions (e.g. during usability evaluations), while in participative role they actively participate in the design process and have decision-making power regarding the design solution (being, e.g. part of the project group) (cf. [12]).

User participation might also be indirect; the HCI literature suggests hiring a group of experts, variably labeled e.g. as usability, UCD, UE or interaction design specialists to 'represent the users' in the development [20]. They will be called HCI specialists in this paper. These specialists are expected to carry out or facilitate the user and context of use analysis, the user task or work practice redesign, design of the HCI solutions and the iterative usability evaluations. [4, 8, 20, 26, 32, 34] In case these specialists are involved in the process, it is interesting to consider also their role, which can also be classified as informative, consultative or participative in the similar sense as is the case with the role of users (cf. [20]). It might be that the HCI specialists are only allowed to act as providers of information (delivering knowledge about the user or general state-of-the-art HCI knowledge to the development) or as

commentators of predefined design solutions (carrying out different kinds of usability evaluations). However, they might also be allowed to be in participative role actively taking part in the design process together with the developers, having decision-making power regarding the design solution [20]. Finally, a part of recent HCI literature suggests that design solutions should be produced solely by the professional HCI specialists, who are to have the power and authority to produce solutions that suit the users (called e.g. interaction designers, cf. e.g. [7, 25, 27]). They are expected to have 'knowledge about the user' and general state-of-the-art HCI knowledge, through which they can claim to be well-equipped to 'represent the user' and produce suitable solutions for them [20].

5 Proposed Model

The traditional decision-making structure of OSS projects depicted as the onion model does not give enough room for HCI specialists. Thus, we propose a model that extends the structure by adding another level of communication and roles (Figure 2). The proposed model is based on the literature described above, as well as our own experiences in real OSS projects.

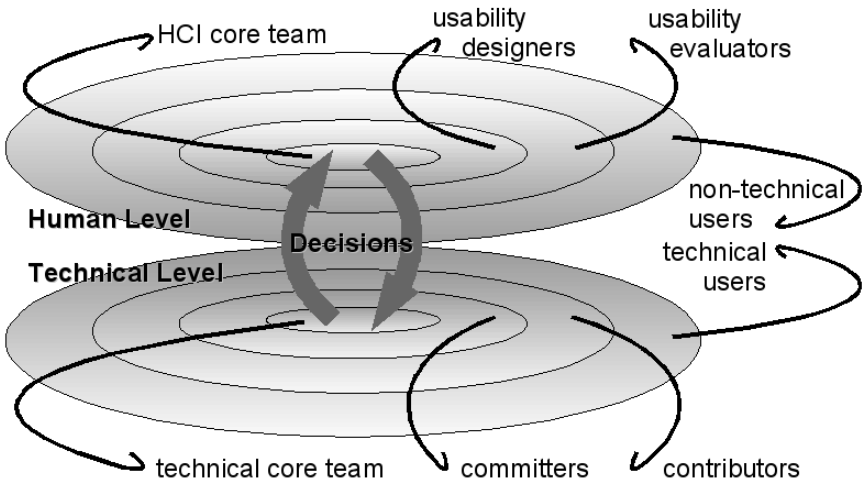

Fig. 2. The proposed model with the technical level and the human level roles

The technical level of the model is based on the traditional organization of OSS projects. The central part is the technical core team who has the decision making power related to technical issues. All feature level changes must be accepted by the technical core team before those are integrated into a release. The next layer is formed by committers who have access to the source code repository. They may fix

bugs and implemented planned features without restrictions. Contributors are participating in the project more randomly. They may offer bug fixes and implement new features in the form of patches, but they are not able to affect the code base without the help of a committer or a core team member. The user layer of this level refers to users who are capable of filling bug reports and have at least some technical knowledge.

The human level of the project structure comprises of HCI specialists and possible application domain experts who may not have deep technical knowledge. The layout mimics of that what constitutes the technical level. Again, the core team makes all important decisions, but in the human level, issues cover user interface and functional requirements, and thus, it is called a HCI core team. Basic principle is that all user interface related changes must be accepted by the HCI core team before the implementation begins or before the possible prototype implementation is integrated into the current mainline product code base. Usability designers have a similar role than committers, but instead of source code their matter is user interface in the form of design, mock-ups, scenarios and similar documents. They can revise the existing features on their own, but major modifications and new functionality must be circulated through the HCI core team before implementation. The next layer in the human level is not directly affecting the product. Usability evaluators gather information on users in the form of user studies before the actual implementation starts, and perform usability tests and other evaluation activities with the actual product. They are also the missing link between passive end-users and the OSS development project. They produce information about the usability of current product, prototypes or plans based on the guidelines, heuristics or feedback gathered from the actual end-user population for the use of the rest of the OSS project. The existence of active users without technical competence is acknowledged in the human level. They have opportunity to participate in the project by communicating HCI related issues with the HCI specialists rather than developers.

The presented roles are based on the typical tasks identified in the HCI literature as well as in OSS projects. Naturally, an individual person can act in several roles in the very same project. For example, a capable developer can contribute both source code and user interface design proposals, or a technically oriented usability evaluator can also write reports on implementation level bugs. In addition, the onion-like structure does not impose hierarchical communication or chain of command, but the increased activity and influence from outer to inner layer in the project. For example, an active user may offer her brilliant ideas directly to the HCI core team, but she cannot modify the user interface design plans like the usability designers and the HCI core team is able to do. Similarly, it might even be convenient that a contributor negotiates user interface issues with the HCI core team without intermediaries. It is also worth noticing that the model does not contain passive users, because although they have an important role as users of the software, they do not directly participate in the development project. When they start to do so, they become active users.

The proposed model highlights technical-oriented and human-oriented thinking, which both have important roles in successful projects, and which both need to be

supported. The idea is that also the language used is different in both levels. In the technical level, technical jargon and programming language syntax is used as in traditional OSS developer intercommunication. Also the actual source code is reviewed and discussed in this level. Respectively, in the human level, the messages should be expressed in natural language together with visualizations. It is recommended to avoid implementation details, but the human viewpoint is preferred instead. This way HCI specialists with less knowledge and interest in technical aspects are not blocked away.

The layers in the human level reflect also the different variations of the HCI specialists' roles in the project. Usability evaluators are in the informative or consultative role based on their activities, while designers are more in the participative role, and the HCI core team is in the decision maker role. However, the overall power of the HCI specialists depends on the division of decision making power between the HCI and the technical core team. It can be adjusted based on the project needs. An extreme example is that all user interface decisions are made by the HCI core team, while the technical core team may choose between pure technical options.

As mentioned, there is a risk that the HCI methods and processes do not fit the OSS development process and philosophy (cf. [3, 5, 6, 39]). For that reason, we try to adjust the model to fit the traditional OSS development context as much as possible. Since the technical level in the model is based on the existing OSS project organization style, the OSS developers should feel comfortable to participate in the project utilizing the proposed model. The basic set of tools, communication methods and working procedures remain the same. The major difference is the interaction between the technical and the human levels. Although it is not certainty that the developers are willing to co-operate with the HCI specialists, the proposed model makes the existence of the HCI specialists visible in the projects, and promotes the given interaction points in the course of a project.

Another problem already mentioned is the lack of support for distributed design in the HCI research [6, 29, 30]. This issue is considered in the model. However, the tools enabling distributed work in the human level depend heavily on the application domain, HCI activities performed, and project maturity. Nevertheless, following the OSS principles, remarkably simple tools can offer significant effects. The HCI core team can be organized around a mailing list just like the technical core team tends to be organized in the traditional OSS projects. Wikis are powerful repositories for textual information, they are able to store also pictures and documents, and can act both as a storing place and a discussion forum for usability evaluation reports and user interface designs. Also the version control system is a potential location to save usability related material, thereby integrating the outputs of the two levels more tightly together. However, it must be recognized that visualizations are more important in the human level than in technical level, of which the main output is the source code. In addition, enabling active cooperation during producing the design outputs is particularly important in the human layer. Therefore, the wikis and

potentially also blogs [30] are to be preferred over the version control systems in the human layer.

6 Evaluation

The proposed model is evaluated in the EU funded scientific and technological project called UMSIC (Usability of Music for Social Inclusion of Children, www.umsic.org). The model will be put into real test during the three years time the project lasts, and empirical data related to the suitability of the model and its improvement possibilities will be continuously gathered.

The UMSIC project aims at developing a mobile time- and place-independent music application that provides an interactive environment for children whilst also enabling them to communicate musically and informally with their peers. The project is run by an interdisciplinary and multinational group of experts, whose fields of expertise are developmental psychology, music education, music technology, music therapy, software engineering, and human computer interaction for children. The target group consists of preschool children who are not able to directly participate in the development of the product. Thus, the HCI specialists are really needed.

The actual software development is done on OSS style under GPL (GNU General Public License) to allow all interested parties to participate in the development, and especially to provide an open and easy-to-access solution for adding new contextual situations and educational contexts. The researchers of the project constitute the permanent structure of the development project, mainly the technical and the HCI core teams, while other developers will change in the course of the project. The workforce will include software engineering students, who participate as OSS developers and HCI specialists. Although the researchers have the final decision making power, developers are encouraged to work autonomously and negotiate only major decisions with the core teams through mailing lists, thus promoting OSS philosophy. The project will also advertise the development in the OSS events and invite everybody who has interest in the subject to join the project in a suitable role.

In addition, the proposed model has already been evaluated on two occasions to gather important feedback before putting the model into use. First of all, it has been presented and discussed in a project meeting, in which domain specialists, HCI specialists, key developers and management of the project were all present. The participants generally agreed with the presented ways of working outlined in the model. However, the following point was emphasized: 1) The model takes into account the HCI issues, but there are also important considerations related to information security and ethical issues in the project. How should these be handled? Should there be separate levels also for these issues?

The proposed model has also been initially evaluated in an OSS seminar, in which around 100 OSS experts and enthusiasts gathered to present their experiences and research results. After the presentation, three additional points were raised related to

the model: 2) People have tried to report usability bugs to OSS projects, but the OSS projects have not taken them into account in any way. How does this model help in this situation? 3) The core team might consist totally of employees working in a certain company. In this situation they will probably make all the decisions from the viewpoint of the benefits gained by the company. How this should be dealt with in the model? 4) Finally, it was emphasized that voting mechanism exists in OSS projects. Wouldn't it be better to rely on them, i.e. to rely on users themselves taking part and influencing the development?

These comments have all been taken into account and the proposed model has been refined, specifically related to the fourth point arguing for allowing more influence for the active end-users. Related to the first point, in many OSS projects these issues probably remain neglected, due to which in this project there needs to be dedicated persons and procedures taking care of security and ethical viewpoints. Simply adding new levels for each new viewpoint may not be sufficient, since it may lead to fragmentation of communication and cooperation. In the future, the model could be extended with these additional tasks introduced as new roles in the human level, for example, but initially our interest is in experimenting with the model in practice focusing on the usability viewpoint.

Related to the second point, we argue that the model should be particularly helpful in this respect. The model changes the decision-making structure so that the HCI core team makes the decisions related to HCI issues. If someone reports usability problems, the core team will have to react to them. And since their interest is in any case in improving usability, they surely will not disregard the reported usability bugs.

Related to the third point, we can only argue that if the core team consists of employees working in a certain company, our model does not change or even try to change the situation. Generally, in IT companies nowadays usability and quality user interfaces tend to be important competitive advantages, due to which one could assume that usability is taken into account also in this situation. One could even argue that it probably is taken into account even more in this situation, since the OSS projects typically don't have interest in usability of their solution, but in the companies the situation tends to be the opposite.

Related to the fourth point, we reconsidered the role of the active users, who might be interested in and capable of reporting usability bugs and voting for issues in the OSS community. Of course, this kind of user involvement is the traditional strength of OSS development [39, 40], which we also want to preserve. In the model, it is emphasized that this kind of users can take part in the development in the HCI layer in a similar ways as the HCI specialists: they can act in the usability evaluator or designer roles as well as inform the developers based on user data gathered. However, we still emphasize the need of HCI specialists for ensuring usability. As mentioned, they are trained for developing usability, unlike most of the developers and users [39, 40]. The HCI specialists are particularly needed in situations, in which the end users (for example young children) are unable to participate themselves. In addition, related to many OSS solutions targeted at non-technical users, the users do

not necessarily even know that they could affect the development, i.e. that there are discussion forums, bug reporting systems and voting mechanisms available - they may not even know that they are using OSS as a part of the product. In this situation, they can't take part, of course. Also, if there has been companies involved in the product development, the users might expect high usability and be unwilling to take part in the development at all, but depend on providers to take care of it.

8 Conclusions

In this paper we have proposed a model to integrate HCI specialists into OSS development projects. Traditional OSS projects are depicted as an onion-like organization model consisting of different kind of stakeholder roles. The participation in a project in that level requires technical competence, such as ability to use bug reporting tools or to read and write source code. Although the need for HCI specialists is acknowledged in OSS development projects, the traditional organizational structure does not leave room for them to participate.

Our proposed solution is to highlight the HCI specialists' roles by adding a new level, the human level, into the onion model. While the technical level consists of technical core team, committers, contributors, and active users, the human level introduces similar roles to the HCI specialists: HCI core team, usability designers, usability evaluators, and non-technical users. These also reflect the various authority positions of the HCI specialists: informative, consultative, participative, or decision maker. The idea is that while the communication methods and tools can be relatively simple and similar in both levels, the language used is totally different. The source code is emphasized in the technical level, but the human level actors prefer natural language and visualizations.

The proposed model is based on the literature on OSS and HCI, as well as our own experiences in real OSS projects. We have also initially evaluated and adjusted the model based on the feedback gathered in a research project and in an OSS seminar of some 100 OSS experts and enthusiasts. The empirical evaluation will be done in an interdisciplinary and multinational scientific and technological project developing a mobile OSS that provides an interactive musical environment for children.

We encourage also others to empirically evaluate the proposed model, and produce detailed reports on the experiences got. Different variations of the model are also welcome. Additional future research topics include the integration of other important viewpoints, such as ethics and security, into OSS development organization.

Acknowledgements

Part of the research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 22456.

References

1. Andreasen, M., Nielsen, H., Schrøder, S. & Stage, J. (2006) Usability in Open Source Software Development: Opinions and Practice. Information Technology and Control 25(3A): 303-312.
2. Apache Software Foundation (2008) Apache HTTP Server Project Guidelines and Voting Rules. Available from: http://httpd.apache.org/dev/guidelines.html. Accessed 30 October, 2008.
3. Benson, C., Müller-Prove, M., Mzourek, J. (2004) "Professional usability in open source projects: GNOME, OpenOffice.org, NetBeans." Extended Abstracts of the CHI2004. pp. 1083-1084.
4. Beyer, H., Holtzblatt, K., Contextual Design (1998) Defining Cus-tomer-Centered Systems. Morgan Kaufmann Publishers Inc, San Francisco.
5. Bødker, M., Nielsen, L. & Orngreen, R. (2007) Enabling User-Centered Design Processes in Open Source Communities. In N. Aykin (ed.), Proc. Human Computer Interaction International, Part I: Usability and Internationalization. LNCS 4559, pp. 10-18.
6. Cetin, G., Verzulli, D. & Frings, S. (2007) An Analysis of Involvement of HCI Experts in Distributed Software Development: Practical Issues. In D. Schuler (ed.): Proc. Human Computer Interaction International: Online Communities and Social Computing. LNCS 4564, pp. 32-40.
7. Cooper, A., The Inmates Are Running the Asylum (1999) Why high-tech products drive us crazy and how to restore sanity, Mac-millan, Indianapolis.
8. Cooper, A. & Reimann, R. (2003) About face 2.0: the essentials of interaction design. Wiley, Indianapolis.
9. Cox, A. (1998) Cathedrals, Bazaars and the Town Council. Available from: http://slashdot.org/features/98/10/13/1423253.shtml. Accessed 22 March, 2004.
10. Crowston, K., Annabi, H., Howison, J. & Masango, C. (2004) Effective Work Practices for Software Engineering: Free/Libre Open Source Software Development. In Proceedings of 2004 ACM Workshop on Interdisciplinary Software Engineering Research, November 5, 2004, Newport Beach, California, USA. ACM Press, 2004, pp. 18-26.
11. Dahlander, L. & Magnusson, M. (2005) Relationships between open source software companies and communities: observations from Nordic firms. Research Policy 34. pp. 481-493.
12. Damodaran L. (1996) User involvement in the systems design process – a practical guide for users. Behaviour & Information Technology 15(16): 363-377.
13. Divitini, M., Jaccheri, L., Monteiro, E. & Traetteberg, H. (2003) Open source processes: no place for politics? In J. Feller, B. Fitzgerald, S. Hissam and K. Lakhani (Eds.), Proc. of the 3rd Workshop on Open Source Software Engineering. Portland, May 2003, pp. 39-43.

14. Feller, J. & Fitzgerald, B. (2000) A Framework Analysis Of The Open Source Development Paradigm. In the Proc. of 21st International Conference on Information Systems, December 10-13, 2000, Brisbane, Australia. Pp. 58-69.
15. Fitzgerald, B. (2006) The Transformation of Open Source Software. MISQ 30(3). Pp. 587-598.
16. The FreeBSD Documentation Project (2008) FreeBSD Handbook. Available from: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/. Accessed 30 October, 2008.
17. Gacek, C., Lawrie, T. & Arief, B. (2001) The many meanings of Open Source. Technical report. Centre for Software Reliability, Department of Computing Science, University of Newcastle, Newcastle upon Tyne, United Kingdom.
18. Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J. & Cajander, Å. (2003) Key principles for user-centred system design. Behaviour & Information Technology 22(6): 397-409
19. Heikinheimo, H. & Kuusisto, T. (2004) The use of embedded open source software in commercial products. In the Proc. of the 13th European Conference on Information Systems (ECIS2004), The European IS Profession in the Global Networking Environment, Turku, Finland, June 14-16, 2004.
20. Iivari, N. (2006) Understanding the Work of an HCI Practitioner. In A. Morch, K. Morgan, T. Bratteig, G. Ghosh, D. Svanaes (Eds.): Proceedings of the 4th Nordic Conference on Human Computer Interaction, Oslo, Norway, October 14-18. Pp. 185-194.
21. Iivari, J. & Iivari, N. (2006) Varieties of User-Centeredness. Proc. of the 39th Annual Hawaii International Conference on System Sciences. IEEE Computer Society Press.
22. ISO 13407 (1999) Human-centered design processes for interactive systems. International Standard.
23. Kujala S (2003) User involvement: a review of the benefits and challenges. Behaviour & Information Technology 22(1), 1-16.
24. Luke, R., Clement, A., Terada, R., Bortolussi, D., Booth, C., Brooks, D. & Christ, D (2004) The Promise and Perils of a Participatory Approach to Developing an Open Source Community Learning Network. In Proc. Participatory Design Conference. New York, ACM, pp. 11-19.
25. Löwgren, J. (1995) Applying Design Methodology to Software Development. In: Proc. Designing Interactive Systems (DIS '95). New York ACM Press, p 87-95.
26. Mayhew, D. (1999) The Usability Engineering Lifecycle: A practitioner's handbook for user in-terface design, San Francisco: Morgan Kaufmann Publishers Inc.
27. Muller, M. & Carey, K. (2002) Design as a Minority Discipline in a Software Company: Toward Requirements for a Community of Practice. Proc. CHI02. New York, ACM Press, p 383-390.
28. Nichols, D., Thomson, K. & Yeates, S. (2001) Usability and Open Source Software Development. In the Proc. of the Symposium on Computer Human Interaction. Pp. 49-54.
29. Nichols, D. & Twidale, M. (2003) The Usability of Open Source Software. First Monday 8(1), 21 p.
30. Nichols, D. & Twidale, M. (2006) Usability Processes in Open Source Projects. Software Process Improvement and Practice 11. Pp. 149-162.
31. Niederman, F., Davis, A. Greiner, M., Wynn, D. & York, P. (2006) A Research Agenda for Studying Open Source I: A Multilevel Framework. Communication of the Association for Information Systems 18. pp. 19-149.
32. Nielsen, J. (1993) Usability Engineering, Boston: Academic Press.

33. Pemberton, S. (2004) Scratching Someone Else's Itch (Why Open Source Can't Do Usability). Interactions January + February. P. 72.
34. Rosson, M. & Carroll, J. (2002) Usability Engineering: Scenario-based Development of Human-Computer Interaction, San Francisco: Morgan-Kaufman.
35. Rubin, J. (1994) Handbook of usabikity testing: how to plan, design, and conduct effective tests. New York: John Wiley & Sons.
36. Twidale, M. & Nichols, D. (2005) Exploring Usability Discussions in Open Source Development. Proc. 38th Hawaii International Conference on System Sciences (HICSS). IEEE.
37. Viorres, N., Xenofon, P., Stavrakis, M., Vlanhogiannis, E., Koutsabasis, P. & Darzentas J. (2007) Major HCI Challenges for Open Source Software Adoption and Development. In D. Schuler (Ed.), Proc. Human Computer Interaction International: Online Communities and Social Computing. LNCS 4564, pp. 455-464.
38. Ye, Y. & Kishida, K. (2003) Toward an Understanding of the Motivation of Open Source Software Developers. In Proc. 25th International Conference on Software Engineering (ICSE), IEEE, pp. 419-429.
39. Zhao, L. & Deek, F. (2005) Improving Open Source Software Usability. Proc. of the 11th Americas Conference on Information Systems. pp. 923-928.
40. Zhao, L. & Deek, F. (2006) Exploratory inspection: a learning model for improving open source software usability. Extended Abstracts of the CHI2006. pp. 1589-1594.