Major HCI Challenges for Open Source Software Adoption and Development

Nikos Viorres, Papadopoulos Xenofon, Modestos Stavrakis,
Evangelos Vlachogiannis, Panayiotis Koutsabasis, and John Darzentas

University of the Aegean - Department of Product and Systems Design Engineering,
Hermoupolis, Syros, Greece, GR – 84100
nviorres@aegea.gr, xenofon@syros.aegean.gr, modestos@aegean.gr,
evlach@aegean.gr, kgp@aegean.gr, idarz@aegean.gr

Abstract. The aim of the paper is to identify and discuss major challenges for OSS from an HCI perspective, so as to aid the adoption and development processes for end-users, developers and organizations. The paper focuses on four important HCI concerns: product usability, support for user and development communities, accessibility and software usability and proposes areas for further research on the basis of related work and own experiences.

1 Introduction

Over the last ten years, OSS (Open Source Software) has experienced widespread recognition and adoption. Supported by the ever-growing internet penetration in both developed and developing countries and the constant momentum-gaining on-line communities, the model has stimulated the interest of not only visionaries, developers, academics and small organizations around the world, but also of large enterprises [11] [30]. The Open Source Definition [18] declares a set of conditions, which can be used to determine whether a software license can be thought of as open source. Briefly, these are related to: the provision of source code, free distribution, the allowance of redistribution of modifications, the protection of the author's original code, the dismissal of discriminations and license-specific conditions.

With respect to the OSS popularity/adoption, there are some very impressive figures. Netcraft's surveys[1] show a sustained market share above 65% for the Apache web server and 33%[2] for Sendmail. GNU/Linux commands greater than 50% market share for infrastructure applications like file/print servers, cache/firewall [12]. In addition, various Open Source programming languages like PHP, Python and Perl, exhibit phenomenal popularity among web developers. There is a variety of reasons for the notified popularity of OSS, including the quality and reliability of OSS products, the rapid release schedules, the reduced cost of OSS development and ownership [11], as well as flexibility and benefits from open standards support.

[1] Netcraft – Web server survey archives 2006.
[2] http://www.securityspace.com/s_survey/data/man.200611/mxsurvey.html

D. Schuler (Ed.): Online Communities and Social Comput., HCII 2007, LNCS 4564, pp. 455–464, 2007.

The widespread recognition and adoption of OSS and its paradigm has attracted significant attention from the user, developer, academic and industry communities. Users are interested in adopting OSS solutions to seek for potential better alternatives to proprietary software, reduce the costs, and even participate in OSS activities and communities. Developers have various incentives [14], including ethical and educational reasons, rapid development cycles, high reusability, and the ability to make a profit while gaining reputation and at the same time get enjoyment out of it (the freedom of choice). The software industry has already found business models to exploit Open Source (Red Hat, IBM, etc.), while on the other hand proprietary software companies are carefully considering the OSS both as a potential business model and as a valid development process (Sun/Java, Microsoft). Finally, researchers and academics are looking into the ways that OSS has influenced software development in general [26] and are exploring means to contribute, evaluate and explain OSS, in terms of its workings, advantages (business and technological), as well as identify potential limitations and inhibiting factors.

Various research efforts have tried to identify challenges and potential limitations of OSS and contribute towards its progress. Some notable areas of concern include, collaboration issues [1] [6], organizational and community issues [6] [5]; [25], security [20] [7], code quality [28] [27] and product quality in general [21]. A group of problems that has attracted limited attention is related to the multidisciplinary field of Human Computer Interaction (HCI), with various concerns and focuses including usability of software, communication, collaboration and more generally interaction issues in the OSS communities. However, an analytic enquiry to the HCI issues related to OSS in an inclusive manner, is of great importance towards the improvement of both the software products and the interaction in OSS communities.

The aim of the paper is to identify and discuss major challenges for OSS from an HCI perspective, so as to aid the adoption and development processes for end-users, developers and organizations.

The paper is structured as follows: section 2 presents the related work and scope. Section 3 presents major OSS challenges from an HCI perspective emphasizing at: product usability, support for user and development communities, accessibility and software usability. Section 4 presents the authors own experiences in developing OSS projects. Finally section 5 presents the conclusions.

2 Related Work and Scope

The penetration and popularity of OSS needs to be considered in relation to the audience that OSS addressees. Firstly, projects such as the Apache HTTP project, Sendmail, BIND (DNS) and various Linux installments can be considered as infrastructure software. Secondly, there is a variety of popular OSS products that qualify as development frameworks (various Apache Software Foundation products), APIs and tools (eclipse), which are addressed to the development communities and whose primary goal is to assist in the design, development and production of software. Last but not least, cases of great success of OSS that are addressed to end users, such as applications and desktop implementations include the Mozilla Firefox, having gained a considerable market share from Microsoft's IE; and OpenOffice,

having attracted interest by individuals and corporations due to its free distribution and support of open standards.

Although the design and development lifecycle of open source projects cannot be classified in a unique and all-encompassing way, due to their number and the diversity in their size, purpose and architecture [25], several traits can be identified. Most such projects make limited use of long established software engineering directives and practices such as formal design procedures, specifications, testing and prototyping of a system [31] [16]. Instead, they often adopt a bottom-up approach that focuses on the development of individual components, while the complete model of the system is formulated along the way, where emphasis is given to technical performance, instead of user interface and information presentation issues [10].

A common challenge for the adoption of open source software lays to a perception of its inadequacy vis-à-vis the technological state of the art. This ranges from Linux sometimes limited support for leading-edge or specialized hardware to the acclaimed reduced functionality of open source programs compared to their close-source competitors. Whether a misconception, a distorted image presented by competitors or a valid argument, this subject, its extent and the reasons behind it are hotly debated topics and positions tend to be biased.

A similar issue is relevant to the interoperability between OSS and mainstream applications. It is a fact that the open source community advocates and actively promotes the use of open standards and open formats, both as from an ideological standpoint and as a means to promote both interoperability and variety in the development of available products. However several proprietary or closed formats dominate today's desktop market and the legislature is often invoked in an attempt to actively exempt open source software from achieving a higher degree of integration and interoperability with the market standards. Apart from creating the ideal landscape for passionate debate between sides, these factors introduce additional challenges for the adoption of open source software by end users. It is possible that end users might encounter difficulties importing and exporting their data when they attempt to migrate to an open source equivalent of their previous application, or to keep both applications working in parallel (such as in the case of OpenOffice support for MS Office documents). Users who adopt the Linux operating system need to check the availability of drivers for their hardware, and expert opinion is not always readily available.

Finally, a matter of significant importance in relation to OSS adoption and development is related to issues that concern HCI's multidisciplinary perspective. The assumption that HCI related problems could constitute inhibitors in both widespread OSS adoption and development is not without merit [17]. Indeed, the success of projects that target end-users, such as desktop-applications, may be correlated to corporate support and industry-practices regarding usability. Typical examples in support of this claim are Mozilla Firefox, which engaged User Interface designers in the community [29], Apple's OS X that employed a proprietary interface to an otherwise OSS and OpenOffice.org that is sponsored by Sun and is backed by the StarOffice HCI experts [4]. Thus, an examination of OSS from the inclusive HCI perspective for the identification of problems and concerns that could affect both the adoption and development of OSS products is deemed plausible.

3 Major OSS Challenges from an HCI Perspective

The identification of the major challenges and concerns of OSS related to HCI – presented here - in combination to empirical findings (discussed in chapter 4) allows us to acknowledge a number of important areas of concern for HCI in OSS development and use. These areas are positioned at various levels of abstraction with respect to OSS, and concern: the usability of the product, the support of OS communities, the notion of accessibility and software engineering usability.

3.1 Product Usability

The approach to software usability by HCI strongly supports the philosophy of user centered design. The notion is that typical end-users should be actively involved throughout the software lifecycle in order to produce software products that best match their requirements. However, this approach to the lifecycle of design and development indicates a potential problematic integration to open source communities and practices, which could inhibit the vision of widespread desktop replacement of proprietary software [22].

Nichols and Twidale [17] perform a review regarding the usability of OSS and discuss how the characteristics of open source development affect it. The major issues they identified include; the absence of usability experts, OSS development incentives which bias towards improvements in functionality instead of usability, the bottom-up OSS approach and the lack of resources. To encounter these problems they suggest various methods, techniques and approaches, such as corporate involvement, the development of automated evaluation methods and usability infrastructures (similar to bug-issuing and resolving tools), and involvement of a variety of stakeholders including end-users, experts and academics.

In [3] the introduction of usability practices to popular OSS products, such as Gnome, OpenOffice.org and NetBeans is discussed. Some of the biggest challenges they discovered are the communication between developers, the confusion regarding the target user audience, process problems (including methodological and problem reporting tactics) and the resumption of responsibility within the community. More recently, [17] attempt to explain how the majority of OSS currently addresses issues of usability with respect to the widely identified challenges. Their reported findings suggest that current systems for reporting usability problems are difficult to use by users, and when they do get used the meaning is often lost due to increased complexity with issues related to the analysis of the problems and proposal of solutions/re-designs. As a solution they propose methods to simplify such procedures and may be summarized to:

- Improving bug reporting, by widening participation through lowering effort, cultural and technical barriers
- Improving the analysis of usability-related problems by the community, by applying HCI theoretical principles, comparisons with established interfaces and engaging the reporter to a discussion
- Supporting argumentation for resolving usability related issues.

3.2 Support for User and Development Communities

In research literature, it is widely acknowledged that the model(s) of collaboration in OSS communities differ significantly from established norms in computer-mediated collaboration [32]. The production of high quality software by OSS communities in a spatially distributed environment, using processes and models that are radically different form typical software engineering principles and methodologies, has been a strong incentive to examine how these communities interact through technology. In general researchers have concluded that the models of collaboration in OSS communities differ significantly from established norms in computer-mediated collaboration.

Mahendarn [15] notified the interactions occurring through code in the OSS communities, indicating a potential problem with examining separately the communication-social perspective and material/results occurring in OSS. In order to overcome these problems, some researchers have adopted an ecological view of the world (OSS community domain) that focuses on the relationships of people and material [19].

From a collaboration perspective, OSS stakeholders coordinate their activities in three information spaces [23] [9] [2], namely:

- the implementation space i.e. the source code
- the documentation space, which consists of the documentation, the web resources that contain information about the project, the information in the code versioning systems and various issue tracking systems.
- the discussion space, which consists of mailing lists (developers-users), forums, emails and blogs (to a lesser extent).

The information spaces are supported by tools that mediate the coordination and communication throughout the projects lifecycle. Predominantly, these tools are OSS products themselves and are provided by the community that embraces the project. With respect to the correlation between the quality of these tools and the technical infrastructure in general to the project's outcome [23] perform an evaluation of their proposed methodological framework for socio-cognitive analysis of collaborative design of OSS (a combination of ethnographic methods and computational tools to analyze interactions) on the Python community and find evidence of an "inverse Conway's Law" [24] that could explain how the technical structure of the software might directly influence the social and governance structure of the project.

3.3 Accessibility

From an ethical point of view Open Source concept is strongly related with accessibility expressed as the "right to access for all". From a design and development viewpoint OSS relation to accessibility is twofold. Firstly, it involves Open Standards, which are used widely to ensure the interoperability of services and maximize access to resources. Secondly, the embodiment of accessibility in OSS design; currently we see a move towards the inclusion of accessibility features into systems, tools and the programming languages themselves as system wide core functionalities (e.g. KDE, GNONE and Java Accessibility). To respond to such a requirement Crombie et al [8]

argue for the need of a unifying and inclusive approach to open source information systems and introduce the so called "(Open Source) Accessible Information Processing" approach. The aim of such an approach is to synchronize various efforts in the accessibility arena and offer them to end-users and business as a 'package'. This package would include explicit knowledge about the exact requirements and implementation of these requirements as practical and re-usable approached that codify accessible information. Their usage by the community will be continually fed back (communication from scratch concept).

The requirement for a unified approach is imperative, considering that to date, despite the large variety of accessibility APIs, standards and software in OSS, the majority of disabled users prefer proprietary software due to availability of accessibility applications in combination to assistive technology compatibility.

3.4 Software Usability

The open source movement has offered programmers a vast wealth of free, readily available tools, several of which are used to develop new open source projects or enhance current ones, including the projects of the tools themselves. In that way, a positive feedback loop is created that leads to the enrichment and enhancement of the entire open source codebase. The influx of new developers and their involvement with existing open source projects is crucial for their expansion and level of maturity [16].

It is thus imperative that software development tools, having such an important role in the maintenance and expansion of the open source codebase, conform to high quality standards, both from a software engineering [28] and an HCI perspective. [17] suggest that the usability of open source software such as compiles and file editors present no usability challenges; however, our experience in extended use of such tools indicates that they present several of the HCI-related issues ascribed to less specialized software. Furthermore, it highlights a few additional challenges.

- The highly modular approach, typical of most OOS development tools, increases the complexity of both installing, using and maintaining them.
- The latest releases of several OOS development tools do not adhere to the backwards compatibility principle, even in the case of minor version upgrades, which occur at a high rate.
- The documentation is often limited or fragmented, spread among several sources (forums, white papers, personal web pages, source code).

According to [24] and our own experience, the quality from an HCI perspective seems to have a significant correlation to the overall quality of an open source project. Therefore, addressing the concerns expressed here in respect to development tools is likely to benefit both the open source community and the end users of OSS.

4 Experiences with OSS Development Projects

In this section, the authors' experience from participation in OSS projects is briefly presented in relevance to the identified HCI challenges.

4.1 E-Class

E-class is a platform developed the Greek Universities Network as a fork of the Claroline platform, an online collaborative learning platform, widely used in academic institutes around the world. It is open source software, based on PHP and MySQL. We have been using *e-class* in the Department of Product & Systems Design Engineering since 2002, as the core of our asynchronous e-learning platform. It is part of our everyday academic life and currently contains 110 courses, 58 registered tutors and 570 registered students. During this time, we have added functionality and improved its user interface to better suit the needs of our students and faculty members. In addition, we have participated in projects related to installing, configuring and further developing the latest Claroline release (1.8) for several universities and technical institutes in Greece. The HCI challenges we encountered by adopting both systems as developers, administrators and end users included:

- The usability of both platforms from the students' perspective has been adequate; a recent internal survey has produced several suggestions for improvement.
- The perspective of tutors varies. E-class has been developed by an institution focused on the Greek academic institutes, and seems better tailored to their needs and way of operation. It is noteworthy that the users' requirements during the demonstration of the Claroline platform to course instructors frequently and consistently included modifications already present in the e-class version.
- The documentation of e-class is adequate, and each page contains a link to online context-sensitive help. As to version 1.8, Claroline's context-sensitive help is limited to a few tools and its Greek translation is very poor. Information in the official Claroline forums is partly in French, posing language barriers to our users.
- Neither platform complies with the W3C WCAG 1.0 accessibility guidelines, although a relevant effort is underway by the Claroline development team. However, similar efforts undertaken by our development team for both the Claroline and the e-class platforms are seriously hampered by the lack of accessibility considerations in the initial design, leading to the need for major code rewriting and even redesign of some components.
- From a development perspective, the source codes of both projects present some interesting (and fairly uncommon) challenges. It appears to be the result of several programmers working during the span of several years with no imposed standards, coding conventions or preplanned architecture. Several variable names and database fields are in French. Parts of the code are of different quality and maturity level. These quirks did not prove major obstacles, but Claroline is a fairly simple system from a software engineering perspective; we see this diversion from established programming best practices as negative and we already witness its impact as we work towards the accessibility of the e-class platform.

From the above case we can see that whenever the HCI principles were taken into consideration (cases b and c), the end result was more satisfying to the users. Furthermore, it is apparent that by involving the user into the design and development process the rest of the challenges may be addressed.

4.2 E-University

E-University (http://e-university.gunet.gr) is a national project that aims to design and develop electronic infrastructures for Greek universities. All these activities are under development as portlets which are going to be hosted from each University's portal. A common academic portal will be a gateway to all Universities portals. All aforementioned portals will be based on a common customizable portal infrastructure that is being developed by the project. A fundamental requirement of this project is accessibility. At this time, e-University subproject responsible for designing and developing the portal infrastructure is at the development phase using Apache Jetspeed 2 open source portal. It is interesting to go through HCI issues that have been faced and relate to the design and development based on such an open source framework:

- First of all, while investigating the possible software framework implementations; even if Jetspeed seemed a very interesting implementation there were some concerns about the status of the project. Jetspeed 2 was on a pre-release stage comparing with the mature Jetpseed 1, but the promising features were tempting. This introduced an increased risk, but the fact that the community was highly active was a decisive factor for its selection.
- An important accessibility problem met to most of open-source portal implementation was the usage of HTML for the layout portlets and portal. Jetspeed-2 makes an improvement by replacing HTML tables with CSS technology.
- Jetspeed 2, by conforming to JSR-168 portlet specifications, makes use of standard CSS classes for the distinctive portlet elements. This approach helps avoid accessibility problems. Nevertheless, these were not enough for an accessible portal. Further improvements were developed by the authors including inter-portlet navigation mechanism and CMS editor improvements.
- Jetspeed 2 introduces decorators and layouts. Both are mechanism to develop portlets and portal user interface following the MVC pattern (i.e. using Velocity, JSP, XSLT)
- Jetspeed 2 provides a flexible profiler and capabilities mechanism that allows the developer to build up personalization and adaptation mechanisms.
- Documentation for the use and development of Jetspeed 2 was a major problem. This was strengthened by the complexity of the architecture itself, but also with the development tools.

5 Conclusions

If OSS products are to exploit their full potential in terms of widespread acceptance, they need to systematically address HCI concerns into their design process. Current research focuses on finding more suitable ways for involving end-users in the development process and supporting constructive analysis and resolution by developers. The support of collaboration (communication, coordination, cooperation) among OSS participants needs to be further investigated, mainly in terms of community (tool) support. The influence of established communities which provide ready-made collaboration support via a plethora of tools, is deemed as very important in ensuring a smooth start-up process and critical mass generation. Lastly, accessibility

is a significant aspect for OSS, which currently from a technological and standards point of view, exhibits unrealized potential. The requirement for a unified approach is imperative, an approach that would both strengthen the OSS's appeal and bring it even closer to its philosophical roots.

Acknowledgements. This work has been partially supported by the EUREKA F-JEWEL project.

References

1. Ankolekar, A., Herbsleb, J.D., Sycara, K.: Addressing Challenges to Open Source Collaboration With the Semantic Web. In: 3rd Workshop on Open Source Software Engineering (2003)
2. Barcellini, F., Détienne, F., Burkhardt, J.M., Sack, W.: Thematic coherence and quotation practices in OSS design-oriented online discussions. In: Schmidt, K., Pendergast, M., Ackerman, M., et Mark G. (eds.) Proceedings of the, International ACM SIGGROUP conference on supporting group work (2005b), pp. 177–186 (2005)
3. Benson, C., Muller-Prove, M., Mzourek, J.: Professional usability in open source projects: GNOME, OpenOffice.org, NetBeans. In: Extended Abstracts of the Conference on Human Factors and Computing Systems, pp. 1083–1084. ACM Press, New York (2004)
4. Benson, C.: Meeting the challenge of open source usability. Interfaces: 9–12 Number 59 (2004)
5. Bergquist, M., Ljungberg, J.: The power of gifts: organizing social relationships in open source communities. Information Systems Journal 11, 305–320 (2001)
6. Bezroukov, N.: Open Source Software as a Special Type of Academic Research. First Monday (4:10) (1999)
7. Bradbury, D.: Documentation dearth undermines open source security. Infosecurity Today, 1(5), 6 (2004)
8. Crombie, D., Lenoir, R., McKenzie, N.: Communication from scratch: towards accessible open source information systems. The First International Conference on Open Source Systems Genova, Italy (July 11–15, 2005)
9. Ducheneaut, N.: Socialization in an Open Source Software Community: A Socio-Technical Analysis. Journal of Computer Supported Collaborative Work. 14, 323–368 (2005)
10. Feller, J., Fitzgerald, B.: A Framework Analysis of the Open Source Software Development Paradigm. In: Orlikowski, W.J., Weill, P., Ang, S., Krcmar, H.C., DeGross, J.I. (eds.) Proceedings of the 21st Annual International Conference on Information Systems,Brisbane, Queensland, Australia, pp. 58–69 (2000)
11. Feller, J., Fitzgerald, B.: Understanding Open Source Software Development, Harlow, Essex, UK, Pearson Education (2001)
12. Gartner note: Open Source Software Hype Cycle 2004, Gartner note #G00120900 (June 2004)
13. Kelly, B., Dunning, A., Rahtz, S., Hollins, P., Phipps, L.: A Contextual Framework For Standards. WWW 2006 Edinburgh, Scotland 22-26 May 2006. Conference Proceedings, Special Interest Tracks, Posters and Workshops (CD ROM)
14. Lerner, J., Tirole, J.: Some Simple Economics of Open Source. Journal of Industrial Economics 50(2), 197–234 (2002) Available at SSRN: http://ssrn.com/abstract=313493

15. Mahendran, D.: Serpents and Primitives: An ethnographic excursion into an Open Source community. Master's Thesis, School of Information Management and Systems, UC Berkeley (2002)
16. Mockus, A., Fielding, R.T., Herbsleb, J.: Two Case Studies of Open Source Software Development: Apache and Mozilla. ACM Trans. Software Engineering and Methodology 11(3), 309–346 (2002)
17. Nichols, D.M., Twidale, M.B.: The usability of open source software. First Monday 8(1) (2005) http://firstmonday.org/issues/issue8 1/nichols/
18. Open Source Initiative: Open Source Definition (v1.9) (Accessed 10/01/2007) http://www.opensource.org/docs/definition.php
19. Osterlie, T.: In the Network: Distributed Control in Gentoo Linux. In: Proceedings of the 4th International Workshop on Open Source Software Engineering, Edinburgh, Scotland, pp. 76–81 (2004)
20. Payne C.: On the security of open source software - Information Systems Journal, 12, 61–78 (2001)
21. Porter, A., Yilmaz, C., Memon, A. M., Krishna, A. S., Schmidt, D.C., Gokhale, A.: Techniques and processes for improving the quality and performance of open-source software in Software Process. Improvement and Practice (Wiley) vol. 11(2) (2006)
22. Raymond, E.S.: The Cathedral and the Bazaar. First Monday, vol. 3(3) (March 1998) at http://firstmonday.org/issues/issue3_3/raymond/
23. Sack, W., Détienne, F., Ducheneaut, N., Burkhardt, J., Mahendran, D., Barcellini, F.: A Methodological Framework for Socio-Cognitive Analyses of Collaborative Design of Open Source Software. Comput. Supported Coop. Work 15(2-3), 229–250 (2006)
24. Sack, W., Ducheneaut, N., Mahendran, D., Detienne, F., Burkhardt, J.M.: Social Architecture and Technological Determinism in Open Source Software Development. International 4S Conference: Social Studies of Science and Society, Atlanta, GA (2003)
25. Scacchi, W.: When Is Free/Open Source Software Development Faster, Better and Cheaper than Software Engineering? In: Koch, S. (ed.) Free/Open Source Software Development (2004)
26. Spinellis, D., Szyperski, C.: Guest Editors' Introduction: How Is Open Source Affecting Software Development? IEEE Software 21(1), 28–33 (2004)
27. Spinellis, D.: Code Quality: The Open Source Perspective. Addison Wesley, Reading (2006)
28. Stamelos, I., Angelis, L., Oikonomou, A., Bleris, G.: Code quality analysis in open source software development – Information Systems Journal, 12, 43–60 (2002)
29. Trudelle, P.: Shall we dance? Ten lessons learned from netscape's flirtation with open source UI development. Open Source MeetsUsability Workshop, Conference on Human Factors in Computer Systems - CHI 2002 (2002)
30. VA Software whitepaper, SOURCEFORFE (2004), Leveraging Open Source Processes and Techniques in the Enterprise (November 2004) (Accessed at October 25, 2006) http://sourceforge.aservo.com/Leveraging_Open_Source_Processes_in_the_Enterprise_-_VA_Software.pdf
31. Vixie, P.: Software engineering. In Open Sources: Voices from the Open Source Revolution. Dibona, C., Ockman, S., Stone, M. (eds.) O'Reilly, Sebastopol, Calif, pp. 91–100 (1999)
32. Yamauchi, Y., Yokozawa, M., Shinohara, T., Ishida, T.: Collaboration with Lean Media: how open-source software succeeds. In: Proceedings of the 2000 ACM conference on Computer supported cooperative work, pp. 329–338 (2000)