Learning and the imperative of production in Free/Open Source development

Evangelia Berdou
Media and Communications Department, London School of Economics and Political Science, e.berdou@lse.ac.uk

Abstract. This paper examines the role of learning in structuring access and participation in F/OS communities. In particular it highlights the challenges and barriers to access faced by new developers and the expectations of senior developers regarding the mindsets and capabilities of new contributors. It is argued that learning in F/OS is inextricably connected with the demand for continuous production. The evidence presented is drawn from interviews conducted with inexperienced and experienced contributors from the GNOME and KDE projects. The author challenges the view of learning as an enculturation process and the paper contributes to the understanding of power relations among established and peripheral members in communities of practice.

Learning forms an integral part of the experience of participation in F/OS projects and underlies many aspects of collaboration. Given their limited resources, F/OS communities make significant efforts to lower the barriers to entry for new developers. Nevertheless, new developers face a number of difficulties which are associated with different aspects of development and participation. The paper draws on doctoral research [1] to highlight the challenges inherent in the learning process in F/OS communities from the perspective of new and senior developers. Theoretically, the paper contributes to a better understanding of power relations in communities of practice.

2 Background to the study

This section situates the argument within the context of existing contributions related to learning in F/OS communities and outlines the theoretical and methodological framework for the study

Learning features as one of the main motives for participation in F/OS, and learning practices and processes, such as peer-review, are also often regarded as constitutive elements of the F/OS development model. Studies related to learning in F/OS fall into two broad groups. The first consists of studies that examine the role of tools and the technical characteristics of projects in the learning process. The second group includes studies that focus primarily on issues of socialization and joining. Examples from the first group include Shaikh and Cornford's [2] examination of Version Control or Concurrent Version Tools (VCT or CVS) and Baldwin and Clark's [3] examination of the role of code architecture in organizing and inviting participation. A representative example from the second group is von Krogh et al.'s [4] paper on 'Community, joining and specialization in open source software development' where it is argued that newcomers who eventually received a CVS account adopted specific joining scripts, and patterns of behaviour involving levels and types of activity necessary to become a community member. Similarly to other studies, such as Ducheneaut [5],von Krogh et al. adopt an oversocialized view of learning, predicated on the idea that communities are built upon consensus, shared values and continuity.

Theoretically, the research adopted the CoP perspective and employs Foucault's notion of relational power to address its deficiencies with regard to understanding the power relations between central and peripheral members.

The CoP perspective was developed by Lave and Wenger [6] to account for forms of learning that take place outside the contexts of formal education, such as learning by doing, and learning-on-the job. CoP are formed mainly through the pursuit of a shared enterprise. The theory suggests that as new members, termed legitimate peripheral learners, adopt the ways and practices of the community they move from its periphery to its centre. The approach has been frequently applied in the study of F/OS.

In their critical overview of the way the CoP perspective has been appropriated, Contu and Willmott [7] indicate that most studies conceptualize learning as a process of enculturation into the shared values and norms of CoP and regard CoP as locales of knowledge management. Consequently, according to these two researchers, the more radical elements of Lave and Wenger's original framework, namely the way factors such as access to resources can restrict access to positions of initial peripherality and potential mastery are underdeveloped.

The Foucauldian notion of relational power [8] is useful in providing a framework for the study of power relations in CoP, because it helps draw attention to the complex interdependencies that form between members in the course of their shared pursuit[1]. This is particularly helpful in the context of F/OS which are predominantly voluntary. According to the relational view, power is neither a zero-sum game where different actors compete for resources nor something that is given or exchanged, but is a force that creates complex dependencies and invites a diversity of initiations and reactions on the part of the people involved in them.

The use of relational power has methodological implications, since it focuses the investigation on the concrete practices and tactics employed by CoP members to establish a distinctive system of differentiation between integrated and potential members.

The data presented in this paper are drawn from 25 individual, semi-structured interviews with a wide sample of experienced and new developers involved in the GNOME and KDE projects. The interviews were thematically and discursively analysed.

3 Empirical Findings

The findings are organized according to challenges and learning requirements from the points of view of new and experienced developers.

3.1 Learning and contributing: the newbies' perspective

The stumbling blocks to participation for potential F/OS contributors can be organized into three broad categories. First, there are difficulties associated with the technical aspects and tools of F/OS development, such as the use of CVS. Secondly, there are conceptual difficulties related to understanding the development process and the architecture of the program, how they are set up, how things fit and how they are expected to be put together. Thirdly, there are difficulties related to how newbies situate themselves in the development process, and the selection of tasks that are appropriate to their level of skills.

Before they can reach the point of fiddling with the code, new developers have to learn how to download (check-out), build and install the program's sources. This will allow them to run the latest, in-production, version of the code, a prerequisite for participating in the ongoing development process. Installing a development snapshot is far from straightforward. Once a newbie overcomes this initial hurdle and writes a patch the question arises of how to submit it in the correct format. A set of rules is needed that will allow community members to build the submitted code along with the rest of the resources. Both these processes, checking out code and checking in code, require not only a degree of familiarity with the CVS, but also a conceptual understanding of how "things are put together". Moreover, the incorporation of a patch into the main development tree depends not only on its technical merit, but also on its conformance with the maintainer's view of the appropriate features, and its compliance with the architecture of the module.

Although the documentation that is provided often gives information on some of these issues, such as how to use the CVS, it frequently fails to provide answers to the more conceptual aspects of development. Moreover, even if it exists and is updated, finding the appropriate documentation is often an arduous task. One of the problems most frequently indicated by interviewees concerned the fragmented character of the documentation and other online resources. Neal (19/10/2004), another newbie contributor, indicated that often the information required was not offered in the form of a dedicated resource, but was obtainable from developers' blogs. According to Neal, blog syndication sites, such as Planet GNOME and Planet KDE, are useful because they centralize development information, despite the fact that developers' entries need to be scanned to separate the social, from the technical aspects of information.

This intertwining of the social with the technical, highlighted by Neal, which is characteristic of CoP, may account for the frequently employed practice of lurking on the project's development mailing lists. The public, archival character of the mailing list and its use as a repository of knowledge makes the posting of a message a non-trivial affair, especially for new developers. As the next section will show, experienced developers usually assign considerable importance to how newbies comport themselves on mailing lists, especially the ones that carry the most important development traffic.

Most of the interviewees considered finding a task that is appropriate for their level of skills as one of the major hurdles to participation. Several interviewees had become involved in the development by first assuming more peripheral tasks, such as translating, or by concentrating on fairly self-contained development tasks such as bug triaging[2]. In certain cases, having tasks or projects explicitly addressed to new developers appears to greatly facilitate participation, not least because their initiators will often assume the role of mentor. Although responsiveness is often seen as being one of the key characteristics of F/OS, getting the right people to pay attention to suggestions and look at work is not straightforward.

Despite, and in some cases because of, these difficulties, many interviewees described their experience of participation and collaboration in F/OS as educational and much more valuable than the formal training that is provided by most computer science degrees. In fact, in many instances, participation in F/OS was framed in terms of vocational training. To summarize, therefore, newbies describe integration as a slow learning process during which they build up their skill sets and their community knowledge and position themselves in the development through their choice of tasks.

3.2 Learning and production: the senior developers' perspective

The discourse of more experienced developers relating to new contributors is informed by a production-oriented view of the development process. This shapes their expectations in terms of the behaviour and performance of newbies and guides their decisions about helping them.

One of the characteristics most valued in new contributors and F/OS developers in general, is self-reliance. Many interviewees stated that new developers often expect to 'have their hands held' and to be assisted every step along the way. The rapid release rate of F/OS development and the fact that many developers work on a voluntary basis makes time a very valuable resource in F/OS. Every minute spent helping a newbie is a minute less on writing code. Combined with the high turn-over rate of contributors is the phenomenon of programmers who say that they want to help but who then disappear, which means that senior developers generally take great care about choosing who to help. One of the first things senior take note of in assessing the potential of new contributors is their chosen entry point in the development and the way they initially present themselves on the mailing lists. Newbies who demonstrate that they have tried to develop their understanding of the project's architecture and have an idea about that tasks that they might be able to

[2] Triaging is a Quality Assurance process that involves confirming good and reproducible bug reports from the projects' bug tracking tools, in order to identify exactly which actions generate faults in the program.

perform are regarded more positively than those who simply ask for general help and guidance.

Senior developers' rules of thumb for assessing the potential of new contributors indicates how very much intertwined are the values of self-reliance, commitment and productivity. Putting in the time and the effort to find things for oneself is an indication of commitment and, at the same time, a prerequisite for sustained participation. Successful information seekers and dedicated learners do not impose on the time and attention of senior developers and the incremental self-relying development of their knowledge, a common characteristic of experienced developers, attests to their potential as productive contributors. Furthermore, seasoned developers usually judge the potential of new contributors very quickly, sometimes even from their first few postings. The way potential contributors introduce themselves to the community is not just a matter of successful 'face-work', a sign of whether or not they have successfully assimilated the behavioural 'scripts' of F/OS development. A newbie's initial postings seem to indicate the extent to which they have already committed themselves to the development process.

4 Discussion & Conclusions

The research indicates that although the importance of helping out new developers is generally recognized and attempts to organize and facilitate their integration are reflected in the existence of mailing lists specifically set up for this purpose and the provision of tutorials and documentation, it is also understood that peripheral participation is something that needs to take place in the background and not at the forefront of development. New developers are generally expected to orient themselves by making do with whatever learning resources are available and gaining a working understanding of the project before seeking the help of experienced developers. New developers who seek help on specific issues having demonstrated an active engagement with the project are generally considered more promising than newbies who ask for general help and guidance. The investigation of the dynamics of cooperation between senior and new developers suggests that the role of learning in F/OS communities goes beyond that of establishing a common framework of shared values, practices and networks of contacts between peripheral and central members. The analysis of the interviews indicates that learning processes are integral to the exercise of power and control. The significant barriers to entry, are viewed by senior developers as necessary elements of a process that ensures the level of commitment and capabilities required of new contributors. These barriers indicate that even access to positions of initial peripherality is structured.

A possible explanation for these two seemingly inconsistent strategies, community efforts on one hand to lower the barriers to participation and experienced developers' strategies for attracting the 'right type' of contributors on the other, can perhaps be found in the inherent tension that exists between the need to attract and integrate capable volunteers and the demands of continuous production. F/OS are not simply communities set up as knowledge management locales; they are primarily communities organized around the production of a complex good, software. A significant differentiating factor compared to traditional apprenticeship contexts concerns the minimum degree of commitment demanded in order to be recognized as a legitimate peripheral learner. In F/OS individuals can contribute as much as they want, to any level they went and when they want. However, the ease of signing up combined with the appeal of being known as a F/OS developer means that there will always be more potential candidates, as indicated by the high degree of turn-over of contributors, than legitimate peripheral members. As a consequence, it seems that the criteria for being recognized as a potentially valuable contributor in F/OS differ substantially from those for offline professional networks and communities of practice where institutional frameworks, formal employment relations, formal accreditation schemes and tighter social networks ensure a certain level of skill and some degree of continuity and commitment.

This paper provides a basis for understanding the role of learning in structuring access and participation in F/OS communities. One of its major limitations is that it does not examine failed cases of legitimate peripheral participation, but focuses only on successful ones. In addition, the study did not take account of the contextual factors of learning and participation, i.e. how the issues of culture, language and the existence of a supportive network might affect legitimate peripheral participation.

The research was funded by the Greek State Scholarships Foundation (IKY). In its later stages the study was supported by the EU funded OPAALS project (contract no: 034824). The views expressed in this paper are those of the author.

References

1. Berdou, E., Managing the Bazaar: commercialization and peripheral participation in mature, community-led Free/Open Source Projects, in Media and Communications Department. Forthcoming 2007, London School of Economics and Political Science: London.
2. Shaikh, M. and T. Cornford, Version Control Tools: a Collaborative vehicle for learning in F/OS. 6th International Conference on Software Engineering proceedings - Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engineering, Edinburgh, Scotland, May 25th, 2004.
3. Baldwin, C. and K. Clark, The Architecture of Participation: Does Code Architecture Mitigates Free Riding in the Open Source Development Model? Harvard Business School Working Paper, Final Version, 2005.
4. von Krogh, G., S. Spaeth, and K.R. Lakhami, Community, Joining and Specialization in Open Source Software Innovation: a Case Study. Research Policy, 2003. **32**: pp. 1217-1241.
5. Ducheneaut, N., Socialization in an Open Source Software Community: a Socio-technical Analysis. Computer Supported Cooperative Work, 2005. **14**(4): pp. 323-368.
6. Wenger, E. and J. Lave, Situated learning: Legitimate Peripheral Participation. 1991, Cambridge: Cambridge University Press.
7. Contu, A. and H. Willmott, Re-embedding Situatedness: The Importance of Power Relations in Learning Theory. Organization Science, 2003. **14**(3): pp. 283-296.
8. Foucault, M., The Subject and Power, in Michel Foucault: Power/ Essential works of Foucault 1954-1984. Vol. 3, J.D. Faubion, Editor. 1982, Penguin Books: London; New York. pp. 326-348.