To Rule and Be Ruled: Governance and Participation in FOSS Projects

Zegaye Seifu, Prodromos Tsiavos

▶ **To cite this version:**

Zegaye Seifu, Prodromos Tsiavos. To Rule and Be Ruled: Governance and Participation in FOSS Projects. 6th International IFIP WG 2.13 Conference on Open Source Systems,(OSS), May 2010, Notre Dame, United States. pp.380-388, 10.1007/978-3-642-13244-5_35 . hal-01056037

HAL Id: hal-01056037

https://inria.hal.science/hal-01056037

Submitted on 14 Aug 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

To Rule and be Ruled: Governance and Participation in FOSS Projects

Zegaye Seifu[1] and Prodromos Tsiavos[2]

[1] University of Oslo, Norway
[2] London School of Economics, UK

Abstract. Free and Open Source Software (FOSS) Development has evoked images of full participation, emancipation and flat organization. Despite such rhetoric, some recent studies and practices reveal the re-emergence of hierarchical structures in one form or another as an almost inevitable aspect of the software development process. The objective of this paper is to investigate, both theoretically and empirically, the reasons behind this reappearance of hierarchy and its impact on the participation patterns of open source projects.

1 Introduction

Since its inception, Free and Open Source Software (FOSS) Development has evoked images of full participation, emancipation and flat organization. However, despite the licensing framework and rhetoric surrounding FOSS projects [7, 10], recent analysis of the software development process reveals a more complex picture: the vast majority of contributions come from a small group of core developers who control the architecture and direction of development [5]. In addition, most participants typically contribute to just a single module, though some modules may include patches or modifications contributed by hundreds of contributors [8].

It seems, thus, that while FOSS as a philosophy and licensing practice invites maximum participation in the development process, in practice, the hierarchical structures re-emerge in one form or another as an almost inevitable aspect of the software development process. The objective of this paper is to investigate, both theoretically and empirically, the reasons behind this reappearance of hierarchy and its impact on the participation patterns of open source projects. More generally, it explores the ways in which governance mechanisms in different FOSS projects play a role in the structuring of participation and contribute to the openness of a project. Our effort is to explore the ways in which governance structures contribute to the deepening and widening of participation in the FOSS context.

A multiple case study approach that links to the conceptual framework allows us to offer the foundations for appreciating the role of participation and governance of FOSS projects. The three cases investigated here are complementary and contribute differently to the different theoretical perspectives support-

ing our analysis. Data for the study was obtained from two main sources i) mailing list archives ii) formal documents and websites of the projects.

2 Analytical concepts

2.1 FOSS Governance

Markus [4] defined FOSS governance as the means of achieving the direction, control, and coordination of wholly or partially autonomous individuals and organizations on behalf of a FOSS development project to which they jointly contribute. She also indicated that FOSS governance can be informal (i.e., enacted through shared norms and social control), formally documented (in written policies and constitutions) or encoded in technology (like mailing lists and version control software).

Related studies so far have been interested in the question of participation in FOSS projects from the perspective of motivational factors: why do talented developers voluntarily contribute? [3]. However, as indicated above, only a minority of developers is responsible for the majority of the contributions. This could be partly attributed to the governance mechanisms of the related projects. Governance structures directly affect motivation and participation of developers, as well as the type and quality of their contributions [9]. The mere availability of source code in open source projects practically doesn't guarantee presence of excess participation and participants. Making progress in analyzing and understanding the nature of collaboration in software development would lead us to develop better understanding of on line and digital communities as participative systems [6].

Markus suggested six elements of formal and informal structures and rules as dimensions and analytical parameters of FOSS governance: ownership of assets (includes intellectual property licenses and formal legal organizational structures), chartering the project (like statements of vision, what the software product should look like, etc.), community management (involves rules about participation and participant), software development processes (including structures that address the important operational tasks of development), conflict resolution and rule changing (including procedures for resolving conflict and for creating new rules) and use of information and tools (which relates to rules about how information will be communicated and managed and how tools and repositories will be used). These categories provide an analytical base for comparing the governance mechanisms displayed in the three projects here.

2.2 CBPP and participation

Commons based peer production happens over the digitally networked environment as large and medium scale collaboration among individuals that are organized without markets or managerial hierarchies. This phenomenon is emerging

everywhere in a decentralized way in the information and cultural production systems. The model was first brought in to light by Yochai Benkler [1, 2]. The fact that CBPP takes place in the ubiquitous digital networks and the wide distribution of low cost physical capital, results in a conducive situation for creativity and mass participation.

Benkler lays out three characteristics of successful peer production and participation: the project/artifact must be modular, the modules should be predominately finegrained or small size and there should be low cost integration mechanisms. These properties determine the number of participants, the scope of varied investments (heterogeneity), the minimal investment required to participate and the simplicity level of integration. He stressed that a project that is at its best with these features can potentially attract many users.

Another key element of Benklers CBPP model as complemented by the recent work of Tsiavos [11] is that of excess capacity, both in the level of the artifact and that of the peer. Excess capacity in the level of the artifact relates to the non rivalerous nature of the artifact that is to be produced: the use of the produced artifact by one user should not hinder the enjoyment of another. This feature of the produced artifact allows maximum dissemination and parallel production and hence contributes to the increasing of the user participation. However, it needs to be complemented by what Tsiavos [11] describes as peer excess capacity, i.e. whoever participates in the development needs to have the skills and the time to do so and these skills and time need to persist over time. Besides the motivational factors indicated in various FOSS related studies, Benkler emphasizes that the more excess time and skills a peer has the more she is likely to be able to participate to a FOSS project.

The legal openness that the licenses ensure and the unrestricted access to a non-rivalerous good, as the software is, are not enough for maintaining a flat organizational structure. The nature of FOSS development is result oriented and meritocratic: what is important is that running code is written and that the best solution is preferred. As a result, the individuals that because of their experience and knowledge are able to contribute the best code are more likely to acquire greater power than individuals that lack such characteristics.

2.3 FOSS and Participation

According to general Intellectual Property Rights theory: in the classic copyright exploitation model, the creator is granted by the law a monopoly right as an incentive to produce a work. In the FOSS model, the focus is not on the work as a whole but rather on the level of the individual contribution. Here, the contributor is not attracted by a monopoly right but rather by other rewards she is herself to define: the reasons why a developer contributes to a FOSS project may be hedonistic, related to self-esteem, peer recognition or may relate to other indirect rewards. In any case, since the rewards are to be identified by the individual itself, the function of the legal regulatory instrument, i.e. the

FOSS license, is to reduce friction for such micro-rewards to smoothly function as motivational factor.

It is also important to note that since from the perspective of producing a final product it is not the individual contributions but rather the accumulative result of all participants effort that is of interest, this sustained effort to reduce frictions and maintain excess capacity does not necessarily focus on the individual. In that sense, the operation of the regulatory means supporting FOSS development can focus on the following three objectives: (a) allowing a single individual to participate repetitively and to make contributions of increasingly greater quantity and quality (deepening participation), (b) on increasing the range and diversity of participants, allowing thus the project to scale, (widening participation) (c) or, ideally, on both.

The principal pure FOSS model assumes that the contributions will be small and resulting from a wide range of participants. Practice and more recent theoretical work however, indicate that such a model is the exception rather than the rule. Small size projects or ecologies of such projects are more likely to be the case. In addition, the emergence of hybrid proprietary and FOSS models are a good indication that the result oriented nature of FOSS is its key characteristic and the licensing model only a symptomatic feature of how such objectives may be best achieved. Since the objective of producing high quality running code can be achieved by deepening or widening participation, organizers of software production are likely to encourage both types of participation in order to achieve the best possible results depending on the specific circumstances surrounding the development of a project.

3 Case Description

Skolelinux is a community-managed Custom Debian Distribution (CDD) aimed at schools. It is by now a Debian sub project to make an overall computer solution and the best distribution for educational purposes. It was initiated in 2001 by a group of four programmers in Norway. For the project in Norway, there is a board consisting of elected members from users and developers every two years. The funding comes from private companies but the government had given most of the project finance at the beginning.

GPL is the preferred license and new contributions are encouraged to stick to it. However any solution that is being considered to be part of the system must live up to the Debian Free Software Guideline (DFSG), which contains the same ideals as in Open source Definition (OSI) and Free Software Foundation. Skolelinux is used in more than 450 schools worldwide - mostly Europe but also in Africa. Regarding the participation of users it is written on their WebPages that the project is community driven, having an ongoing exchange between users and developers.

Varnish is hybrid open source software released under the revised BSD (Berkley Software Distribution) license. The project is handled by a company

called Linpro in Norway. Technically, the Varnish software is an HTTP accelerator on web servers. Most web sites or content management systems (CMS) present dynamic web pages consisting of a number of different elements. Combining these elements is both time-consuming and CPU intensive, and the process is repeated for every individual user, even though the content is often identical. Thus, Varnish temporarily stores the most frequently requested pages in cache memory and effectively presents these pages from cache. As a consequence, users are offered improved services, and server requirements are reduced by a huge percentage.

It was started in early 2006 and all the accompanying tools in the project are open source Subversion, TRAC, Mailman and GNU author tools. Linpro run its business by giving services to customers and through sponsorship. From the mailing list it is possible to trace and tell that the community is comprised of individuals all over the world. Currently, there are up to 200 people registered in the mailing list of varnish.

HISP is a globally distributed open source software development which was initiated in South Africa around 1994 and is based on collaboration between academic institutions, health authorities, and private organizations. The goal of the project is enabling south-south and also south-south-north collaboration. The funding comes from various governmental and non governmental donors, though mainly the Norwegian Agency for Development (NORAD) and the European Commission (EC).

The project develops District Health Information System (DHIS), a for collecting, processing, and analyzing health information for health administration purposes. DHIS 2.0 (and upward versions) is a web-based software package released under the BSD license. It is developed using Java frameworks and supported by other open source tools. It has been implemented in many developing countries in Africa and Asia. The HISP project in general involves academicians from universities doing action research through development and implementation. Thus the DHIS software gets the benefits in terms of resources from the academic environments.

4 Findings and Discussions

4.1 Overall picture of participation in the three projects

The development stage and history of the three projects obviously is different. This also determines the amount of data available in their mailing lists and even impacts the number of participants that each could have. We are aware of the fact that the type and purpose of the software also determines the pool of participants. Instead it is actually the trend that we wanted to show here and that helps most to get a comparative view. Accordingly, we considered the data of all the projects since their respective starting period. Table 1 summarizes

the period, number of messages, number of threads, the number of participants and the size of data considered for each project.

Table 1. Overall picture of participation in the three projects

Project	Year	#of Messages	#of Threads	Size(KB)	#of Participants
Varnish	Feb 2006-Dec 2009	1224	340	908	82
Skolelinux	Jan 2002-Dec 2009	15548	6344	66000	170
HISP	Nov 2008-Dec 2009	3984	2018	9000	74

4.2 Governance mechanisms of the three projects

Based on the analytical framework discussed in section 2, the governance mechanisms of the three projects were analyzed. Table 2 presents their similarities and differences. In all the three cases we notice a relevant consistency between the different modalities of governance and regulation used in order to achieve the desired modes of production. For instance, as in Benklers original model the individual identification of incentives appears to be complemented with a licensing scheme reducing barriers to participation. Thus the dispersed and minimal excess capacity of multiple individuals may be collected and organized. The investigation of the social norms reveals some interesting patterns. In the

Table 2. Dimensions of OSS governance mechanisms

Rules	Vanish	Skolelinux	HISP
Ownership of assets	company-manageed; it is BSD licensed	Community-managed; run by a board of elected volunteers; any OSS license	Developed by academic institution supported through action research; BSD licensed
Chartering the project	Governed by rules of the owning company and share its vision	The vision is promoting freedom and sharing, starting from schools; governed by the Debian charter	Intends to promote south-south-north cooperation and finding ways of supporting the developing world
Community management and development process	Open for anyone interested; commit patches under the supervision of the core developers	Open for anyone with outright privilege to make changes	Open for anyone with outright privilege to make changes
Conflict resolution	No written rules; on discussion in the mailing list with lead of the core developers	Negotiation and voting	Discussions through lists and in person
Use of information and tools	Information is openly accessible; use open source tools	Information is openly accessible; use open source tools	Information is openly accessible; use open source tools

Varnish case, there are no clear procedural or decision taking rules. However, the decisions are made and the guidance is provided de facto by the core developers who are sponsored by the companies driving the development. In like manner, in the HISP project a similar situation emerges: the core developers are Oslo University staff and research students and they are the ones driving

the process and taking the decisions in the absence of any clearly stated rules. In the Skolelinux case, there is a flatter structure with more well defined procedures. Where a kind of hierarchy emerges this is the result of the meritocratic process through which contributions are made. It is not that in the case of HISP and Varnish a meritocratic approach is not followed, but it is the sponsoring organizations behind the development that influence the development direction and effectively control the whole process.

This is a very important finding as it indicates that the control structures are not to be found in a first but rather in a second layer. In the first layer all three project resemble in terms of legal and technological structure and in that sense they all appear as meritocratic. However, in a second layer, there are substantial differences related to the time and expertise sponsoring provided by extra-FOSS organizations. This in turn has a direct effect in the formation of the social norms that will then operate as the main day-to-day governance mechanism for the FOSS projects.

4.3 Participation patterns and governance

In the cases of Varnish and HISP, we observe hybrid models and it is extremely helpful to see where this imbroglio-like nature lies. In our opinion the hybridity of the model is in the way excess capacity is handled: in these two cases, the capacity of the core developers is sponsored in a traditional fashion, though without using a copyright contract, since there is no reason to restrict access to the source code. Linpro pays for the time of the core developers that make the maximum contributions, while the rest of the participants play a complementary role making contributions that require only minimal excess capacity and for which the classic CBPP model is applicable. Similar is the situation in the case of HISP. The core developers excess capacity is sponsored by Oslo University, while the rest of the developers make marginal contributions requiring again little excess capacity.

On the contrary, in the case of Skolelinux, there is no clearly seen equivalent centralized and focused sponsoring and hence the contributions are more evenly spread and the decision making process is much more democratic. This last observation brings us to the issue of social norms formation. In the former two cases the prevailing social norms assign particular relevance to the core developers and there are no strong and detailed community norms. In the Skolelinux case, on the contrary, the decision making process is much more democratic and inclusive.

The interactions between different governance layers are partially only reflected in the participation patterns. In the Varnish and Skolelinux the participation falls dramatically after an initial phase of wide participation, while in the HISP case we see a different pattern. The changes in participation cannot be attributed merely to the governance structures. The maturity of a project, for instance, plays a crucial role: as the project matures, the original easy to be identified and solved problems give their position to more difficult problems,

thus requiring greater levels of capacity and hence limiting participation only to the more knowledgeable and dedicated participants.

The three cases, here, though superficially substantially differ between each other, a second reading present some commonalities as to how the problem of excess capacity is addressed. In the Varnish case, as the project matures, the interest becomes not one of widening but rather of deepening the participation. Since the core developers are being sponsored by Linpro the problem of excess capacity is solved by ensuring that they will keep developing software seeing the community participation as added value to work that is already being conducted by the company. Here, the incentive of Linpro to widen participation is not big enough to lead to any specific measures so that non-paid professionals are able to increase or even maintain the quantity and quality of their contributions.

In the Skolelinux case, while there is an incentive to sustain quality and quantity of contribution, there is no direct ability to sponsor the time and expertise of participants and hence the project inevitably shrinks in size. However, the remaining participants increase the depth of their participation by being able to make more substantial contributions and sustaining a steady number of participants.

Finally, the HISP case is possibly the most interesting one, as it takes a series of conscious steps in increasing the excess capacity of the participating individuals in order to be able to contribute to the project and this is reflected in the number of participants as the project matures. The main mechanism employed by HISP is the use of an educational network that operates as an additional overlay on the FOSS network that supports both financially and educationally the participants. Also, by employing more a more modular and thus efficient software architecture and bug reporting methods, it allows the capturing of even finer granules of excess capacity, thus increasing the participation both in width and depth.

5 Conclusion

Traditionally, FOSS has been associated with openness and participation. Organizational studies have indicated that there are hierarchical elements present in the structuring of the FOSS development process, but have not provided a comprehensive account of why this is the case. In this paper we have presented a first account of why this is the case, indicating, however, that even where participation seems to be reducing its scope, it may be increasing its depth. Wherever hybrid models appear, these are the result of the need to sustain capacity that could not be attracted using only the FOSS model and to achieve value added corrections that could not have been achieved through a classic proprietary copyright management model. These models may lead to a reduction in participation, where there is no provision for increasing the excess capacity of the peers. For these reasons, hybrid network-CBPP models seem more likely to support both widening and deepening of participation in FOSS project.

References

1. Benkler Y (2002) Coase's Penguin, or Linux and the Nature of the Firm. Yale Law Journal 112:369
2. Benkler Y (2006) The Wealth of Networks: How Social Production Transforms Markets and Freedom. Yale University Press, New Haven and London
3. Lerner J, Tirole J (2002) Some simple economics of open source. Journal of Industrial Economics 52
4. Markus M (2007) The governance of free/open source software projects: monolithic, multidimensional, or configurational? Journal of Management Governance 11:151–163
5. Mockus A, Fielding R, Herbsleb, J (2002) Two Case Studies of Open Source Software Development: Apache and Mozilla. ACM Trans. Soft. Eng. Methods 11:309–346
6. Nakakoji K, Yamada K, Giaccardi, E (2005) Understanding the Nature of Collaboration in Open-Source Software Development. Proceedings of Asia-Pacific Software Engineering Conference, IEEE Computer Society
7. Raymond E (2001) The cathedral and the bazaar : musings on linux and open source by an accidental revolutionary, (Rev. ed.) O'Reilly, Cambridge, Mass
8. Scacchi W (2007) Free/Open Source Software Development: Recent Research Results and Emerging Opportunities. Proc. European Software Engineering Conference and ACM SIGSOFT Symposium on the Foundations of Software Engineering
9. Shah S (2006) Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development. Management Science 52:1000–1014
10. Stallman R (2002) Why Software Should Not Have Owners," in: Free Software, Free Society: Selected Essays of Richard M. Stallman, J. Gay (ed.). GNU Press
11. Tsiavos P, Korn, N (2009) Case Studies Mapping the Flows of Content, Value and Rights across the UK Public Sector. Joint Information Systems Committee, London