How Can Open Source Software Projects Be Compared with Organizations?

Remo Eckert

How can Open Source Software Projects be compared with Organizations?

Remo Eckert

University of Bern, Switzerland
remo.eckert@iwi.unibe.ch

Abstract. The existence of a community plays a central role in the development of Open Source Software (OSS). Communities are commonly defined as a group of people sharing common norms or values. The common interest of an OSS project is obvious: to develop software under an OSS license. When we look at the rather general definition of a community, we see that there is a similarity to the term 'organization'. This paper draws parallels between OSS projects and the general elements of an organization and shows the different elements comprised in an OSS community: people, organization and assets. Each of those elements is enriched with examples from different research in the corresponding OSS research stream and provides a broad overview of the elements of OSS projects. With the help of this comparison, research on OSS can be made more focused and aligned with organizational research.

Keywords: Open Source Software, OSS Governance, OSS Framework.

1 Introduction

The phenomenon of OSS has attained much attention over the years. In the academic literature, different aspects of OSS and its development have been examined and discussed. Quite often, however, the terms used to describe the different phenomena around OSS are not exactly defined and even more the relationship between different concepts is not clarified. For example, OSS research often uses various different concepts of a collective that works together to reach a common goal, such like community, project, organization, or foundation. This results in a situation where it is not always clear what is exactly meant with the concepts, what do they comprise and how they relate to each other.

The development of OSS takes place in an OSS project. By creating a three-phase model, de Laat [1] describes the structural evolution of an OSS project. In phase one, governance is spontaneous and explicit coordination and control are non-existent. Phase two introduces internal governance with formal tools, e.g. division of roles, training, modularization or decision-making. This enables an OSS project to be governed internally in order to increase efficiency and effectiveness as the community grows. Eventually, in phase three, if the OSS project is successful and both companies and

other organizations wish to participate, there is a need for institutionalization (a legal entity such as a foundation) to involve outside parties such as organizations [1].

Governance within OSS projects has been widely discussed for many years [2–4]. One frequently used definition of OSS governance is "the means of achieving the direction, control, and coordination of wholly or partially autonomous individuals and organizations on behalf of an OSS development project to which they jointly contribute" [5]. In order to better understand what governance of an OSS project is, one needs first to understand the different elements of what is to be governed. However, there is no research that attempts to explain those different elements of an OSS project. Thus, there is a need for studies that contribute to a better understanding of the different elements comprised within an OSS project. Therefore, this paper attempts to answer the following question: *What are the different elements comprised within an OSS project?* To answer our research question, we have developed a framework in which OSS projects are compared to the elements of an organization.

The remainder of this paper is structured as follows: Chapter 2 shows the elements of an organization because we perceive an OSS project as an organization in the most generic sense. Chapter 3 adapts these elements to OSS projects and explains the different elements of an OSS project. Chapter 4 discusses the results and the implications thereof for both theory and practice.

2 The Elements of an Organization

An organization is defined as an entity comprising multiple actors with a collective goal [6]. According to organizational research, governance combines various mechanisms to encourage people to do things that align with the organization's goals [7].

According to Luhmann [6], there are three characteristics that highlight the organization: first, an organization can decide which people are part of it and which are not. The organization can define restrictions and rules; failure to observe these rules can result in exclusion. Second, organizations have goals and the decisions an organization takes are oriented around these goals. Generally, organizations have several processes, which can be structured either in management processes, in core processes or in supporting processes. Core processes are central for an organization to earn money, whereas management processes structure an organization to achieve those core processes. Supporting processes are necessary to run the core processes, but are not central to an organization [8]. Third, organizations have hierarchies, which regulate the position of members within the organization. Processes and hierarchies enable an organization to coordinate its people. Processes and hierarchies, both formal and informal, therefore represent mechanisms of governance to align the behavior of people according to organizational goals [7].

Besides people, common goals, roles, rules and structures, most - if not all -organizations are in need of assets. Assets are tangible or intangible goods and can be owned or controlled to produce and have a positive economic value. Moreover, they can be converted into cash [9]. From an accounting viewpoint, an asset is a resource controlled

by an entity as a result of past events and from which future economic benefits are expected to flow to the entity. An asset can be tangible or intangible [10].

The establishment of a legal entity helps to protect an organization from various threats such as liability. Fig. 1 combines Luhmann [6] and the accounting viewpoint and shows the different elements of an organization. Although Fig. 1 implies a well-defined structure, each element is closely interlinked with the other elements.

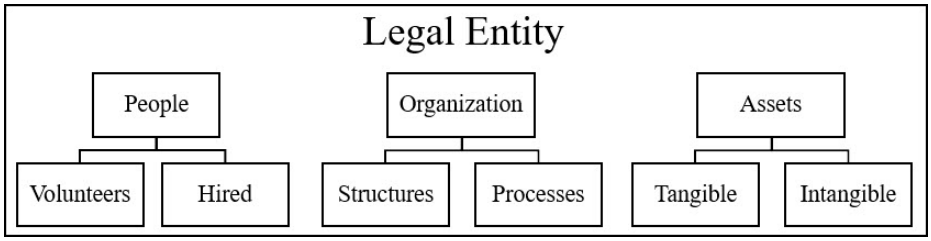

Fig. 1. Organizational Framework.

3 Organizational Framework of OSS Projects

In the following subsections, we explain how each of the corresponding elements of people, organization and assets can be understood in regards to OSS projects. The following subsections show that the similarities between an organization and OSS can be structured according to our organizational framework.

3.1 People

OSS projects are associations consisting of people who come together virtually in pursuit of a common goal [11]. The motivation to contribute to an OSS project differs: contributors to an OSS project can either be paid by an employer, or are volunteers. In general, their reasons for contributing to OSS projects can be categorized as either intrinsic or extrinsic motivation [12]. An action is extrinsically motivated when it is performed in order to obtain some separable outcome, whereas an intrinsically motivated action is carried out for the mere interest in or joy of performing it [13]. However, the motivations behind employees and volunteers contributing to an OSS projects differ [14]. A developer's "itch worth scratching", as stated by Raymond, might be not as strong for a paid developer as for a volunteer [15]. Tasks such as project design, coordination, testing, documentation and bug-fixing are usually less attractive for volunteers and could therefore be carried out by hired people to ensure that these tasks will be done properly [16]. We therefore distinguish between hired people and volunteers.

Volunteers. The involvement of a community in an OSS project is a vital factor for the success of the project because the community promotes the project and its development [17, 18]. Therefore, attracting and gaining volunteers for a new OSS-project is one of the main focus of community building. The community can ease the way in which new volunteers can join the community by defining guidelines, compiling mailing lists and

wikis and answering project-related questions [19]. Moreover, the software quality itself may increase the success rate of attracting new members, whereas several methods of enhancing the quality of the code exist, ranging from code refactoring to documentation [20]. On the technical side, increasing modularity of the source code is one incentive for attracting new developers [21]. Another way to attract more volunteers to an OSS project is to use an issue tracking system [22]. OSS projects typically have an open issue tracker where developers and users of the software can report bugs and feature requests [23]. With the help of issue trackers, potential new contributors can get in touch with the existing community. In their literature review, von Krogh et al. [12] distinguish between intrinsic, internalized extrinsic or extrinsic motivations. As stated by von Krogh et al. [12], some motivations are by definition extrinsic, but could be internalized by developers so that they are perceived as self-regulating behavior rather than external impositions.

Hired Contributors. A high number of developers are paid by an employer for their OSS efforts [24]. In a study by O'Mahony and Bechky [25], 63% of respondents were paid by a corporate sponsor. In the GNOME project, tasks which are usually less attractive to volunteers, such as project design and coordination, testing, documentation and bug-fixing are carried out by employees [16]. People who are paid to contribute to an OSS project may be paid directly by an organization which benefits from the developed software. For example, Red Hat will make more money on support if Linux is used more. Another example is Intel. It will sell more semiconductors if an operating system is free, and a computer therefore costs less. As a result, such companies hope to benefit from allocating their own employees to an OSS project [26, 27]. Berdou [28] distinguishes between free sponsorship, clear mandate, OSS-friendly jobs and sub-contracting. In free sponsorship, developers receive no clear instruction from their employer about what they should work on. For the most part, they are former volunteers, who are expected to work on more or less the same things they used to. In contrast to free sponsorship, those who have a clear mandate from their employer, are told what they ought to work on, such as integrating different aspects of the project into company products, or building on their projects' platform to create commercial applications. OSS-friendly jobs are jobs where people are expected to develop proprietary software, but are also allowed to spend part of their time working on OSS projects. Their terms of work can be formal or informal and resemble part-time free sponsorship. In sub-contracting, meanwhile, people are paid to solve a problem or develop a specific application. This form could also include bounty programs (cash reward offered for development) and self-employed developers. However, people can also be paid by an OSS foundation, as it is in the GNOME Foundation, where the executive director and an administrator, as well as other contractors, are paid via membership fees to accelerate community growth and sustainability [29].

3.2 Organization

Some OSS projects, especially bigger ones, have formal membership rules and agreements, such as the membership fee and bylaws with different roles and functions [19]. Bylaws are rules established by the community to regulate itself in a structural and in a

procedural way. As an example, the bylaws of the Eclipse Foundation regulate the overall purpose of the community, the powers and duties of the various roles within the community, how and when members are elected and how meetings are organized. Moreover, the Eclipse Foundation bylaws explain how decisions are made, explain the tasks of the different committees, councils, boards and the different forms of membership [30]. Contributions not only help the software to evolve, but also redefine the role of the contributors, thereby changing the social dynamics of the community. Consequently, project leaders and core members should focus not only on the evolution of the software itself, but also on the creation of an environment and culture that fosters and encourage new members to move toward the center of the OSS community. This can happen by means of both formal and informal mechanisms. Such mechanisms allow developers to work independently, by encouraging or helping other developers to work in ways that are expected by the community [31].

Structures. Relationships with external groups, leadership and control are common sources of conflict in an OSS project. In the worst case, these can even lead to a breakdown or a software fork where a subgroup of contributors develop their own version of the code [32, 33]. A structure regulates the coordination efforts between different actors. Although there is no strict hierarchy in some OSS communities, their structure is not completely flat. According to [34], roles and their associated influence can be earned through contributions to the community. The resulting informal community structure, called the “onion-model” can be depicted as layers, where the roles closer to the center (e.g. the project leader and core members) have a greater influence than the roles in the outer layers (e.g. readers and passive users). The roles are not fixed and can change over time, depending on the contributions of the community members involved. The processes are in many cases more informal than formal, anyone can join and the relations and roles change over time [35]. Roles do not imply authority, but instead responsibility. Authority, as an example in the NetBeans community, is based on reputation and respect [31].

O’Mahony and Ferraro [36] show how a community uses a formal bureaucratic basis of authority to reinforce its meritocratic norms. They show how an OSS project designed a governance system that combines a constitutional basis of authority with democratic mechanisms to ensure control by the majority. A governance system shapes the way in which project-wide decisions are made. Moreover, governance structures ensure that a project could survive a change in leadership or crucial positions within the project. The need to coordinate member activities and integrate their contributions necessitates a structure. Analyzing the Debian project, O’Mahony and Ferraro [36] distinguish between four different forms of governance, where the community develops from an informal to a formal structure. In the Debian project, the de facto governance worked well for the first five years. There were no formal means of governance. In the designing governance phase, a formal definition of roles, rights and responsibilities was established. The resulting governance system embraced two elements - the formal positional authority and the limitation of that authority through democratic means. In phase three, implementing governance, the new formal governance structure accepted by the community, was implemented. Candidates nominated themselves and the community

could give them their vote. In phase four, stabilizing governance, the leadership forms began to reach settlement.

Processes. Like an organization, an OSS project also has various processes, which can be structured in core processes, management processes and support processes.

Core processes. While many OSS projects are built by a small number of individuals, in large projects, a significant number of individuals and firms contribute to the project [5, 27, 37]. Formal rules about the development process ensure that operational tasks, such as requirements elicitation, assignment of people to tasks, release control, etc. are organized [5]. The Eclipse development process, for example, describes the principles upon which the process should rely: openness, transparency and meritocracy. Moreover, each project is supposed to make a project plan available to the community at the beginning of each development cycle (for each major and minor release) [38]. Another core process is requirements engineering. German [16] illustrates how several leaders in the GNOME project provide a list of requirements that the system should satisfy and reference applications the project should replace. However, not all requirements are an output of the leaders. Requirements for a new module or components are born from discussions in mailing lists or on an issue tracker [22]. Requirements can be formulated in a vision developed by community leaders; by imitating the features from reference applications; by discussions in a mailing list; by providing a prototype with the implemented requirements to work on; or by a post-hoc requirement where the requirements are unknown to the rest of the community and are fulfilled by the developer seeking those requirements. What those methods have in common is that they are usually informal and prioritized by the leaders of the project, maintainers of the module or by a foundation [37].

Management processes. OSS projects, especially larger ones, exhibit formal structures that can be described in bylaws. German [16] describes how the board of directors in the GNOME community is elected. The board of directors is democratically elected by the rest of the foundation members. In addition, the community has a rule that restricts the power of a single organization by imposing a maximum number of board seats. Moreover, board members must represent the interests of the GNOME community, not the interests of their own organization. This rule, as stated, has already been enforced several times in the past.

Supporting processes. The developed software asset needs to be protected from lawsuits from the legal entity that owns it [39]. Therefore, the OSS community needs to ensure that contributions do not infringe third-party IP-rights. For example, a contributor agreement ensures that the OSS community cannot be subject to a firm's ownership claim and is therefore seen in some OSS projects as a precondition to any code contribution [25]. Another example of a supporting process is the funding process. Since there are different forms of memberships and fees, the Eclipse Bylaws define which people have to pay for becoming a member and how much [30]. The secretary is responsible for invoicing membership fees and collecting fees, if necessary with a dues notice. The annual report, which is publicly available, provides an overview of the financial situation of the Eclipse community in terms of revenues and expenses. This report, which encompasses far more than just the financial situation, also constitutes a supporting process.

3.3 Assets

We distinguish between tangible and intangible assets, which are described in detail in the following.

Tangible Assets. With respect to the development of software, the availability of an IT Infrastructure is an important aspect. Gutwin et al. [40] found that distributed developers do need to maintain awareness of one another, more specifically, both a general awareness of the entire team and more detailed knowledge of the individuals they plan to work with. The main mechanisms for maintaining this awareness of who is involved in the project and what their activities are, are text-based communication tools that are commonly used in OSS project, e.g. mailing lists, wikis or text chats. For specific awareness, such as people's expertise and activities, an operating IT Infrastructure with a decentralized version control system such as GIT [41], bug trackers for submitting bugs and feature requests, e.g. Bugzilla [22] and mailing lists [5, 36] are needed. Collaboration among the participating members takes place with the help of these tools and simplifies the effort required for distributed software development [16]. As recommended by German [16], communication should be carried out via a variety of tools. All such tools, including the servers they run on, represent IT Infrastructure [16, 42].

If an OSS project wants to obtain contributions, it needs to market itself. This includes hosting a website with the published source code of the OSS project. Therefore, a webserver is required. As an example, a committer must have access to the latest code base in order to insert changes into that base [43]. Normally, this is done using a decentralized version control system (DVCS) such as GIT, which facilitates collaboration among various developers [41]. However, a DVCS and a website need to run on a server with guaranteed Internet access. As described by German [16], at the level of community IT Infrastructure, servers as well as bandwidth are required to communicate and share collaborators' progress. As an example, the GNOME community relies upon donations from the Autonomous University of Mexico and other organizations that provide its IT Infrastructure [16].

Because the majority of work in an OSS project is performed by globally distributed individuals, face-to-face meetings are rather rare. However, they help to better communicate and resolve potential conflicts [40]. Consequently, infrastructure in the form of rooms for meetings and an internet connection can help to reduce potential communication problems. Moreover, if the OSS project has a legal entity, its address can be used for corresponding purposes.

Intangible Assets. Although OSS does not fully meet the conditions to be included as an asset in financial reports [44], it can be protected in different ways, such as intellectual property rights (IPR), trademarks and brands. As stated by Fitzgerald [45], trademarks or brands are alternative mechanisms to protect IPR in addition to the license itself. A trademark can protect the OSS project's reputation by preventing other projects using their name or brand (e.g. a software fork with other goals than the main project). The goal of a trademark is to prevent customers from confusion as to the origins of the product or service [46]. A common practice among OSS foundations is to own the copyright of the source code and related texts, as stated by Riehle [47].

However, a legal basis for a growing community, especially when firms are involved, is necessary. Firms are reluctant to donate code to a project without transferring responsibility for it. Without a legal owner, firms hesitate to donate code and transfer responsibility for future maintenance. The establishment of a foundation offers firms a legal entity to which ownership can be transferred [16, 25]. IPR are better defined and more defensible when owned by a single legal entity as opposed to various individuals. Having a single and central copyright holder means that it holds the asset and therefore can protect it more easily compared to the situation when hundreds of contributors hold the individual rights to their parts. Furthermore, a legal entity ensures that volunteer contributors are protected against individual liability, and can enter into agreements collectively and protect their code, trademarks, licenses and brands on their behalf [39].

4 Discussion

Our organizational framework compares OSS projects with the typical elements of an organization by showing the different elements exhibited by an OSS project and comparing these with the elements of an organization. According to our framework, the three main elements of an OSS project are people, organization and assets. Each of those elements is enriched with examples from different research in the corresponding OSS research stream and provides a broad overview of the elements of OSS projects.

The people dimension refers to the concept of a community. Contributors to an OSS project therefore can either be hired or volunteers. OSS projects are associations consisting of people who come together virtually in pursuit of a common goal [11]. Tasks such as the project design, coordination, testing, documentation and bug-fixing are less attractive for volunteers and could therefore be carried out hired staff to ensure that these tasks are done [16]. Due to the fact that, in some OSS projects, more than half of the contributors are paid by a corporate sponsor [25], we see that OSS governance is becoming increasingly important. A key aspect of OSS governance is therefore managing the community [5].

What we call the organizational dimension as well as the asset dimension, can be compared to the concept of governance. Similar to Markus [5], we see formal and informal structures and norms as one of the main elements of OSS governance. Different projects have a varying degree of formalism, with some lacking any formal descriptions at all on how decisions are made. However, there are a number of OSS projects with formal rules and agreements, such the bylaws, with different roles and functions [19]. When firms participate in an OSS project, the degree of formalism may increase [25]. Similar to Markus [5], we see ownership of assets as one of the main areas of OSS governance. Although OSS does not fully meet the conditions to be included in financial reports as an asset [44], OSS projects often possess other assets, such as IPR, trademarks, brands or IT Infrastructure [16, 45]. However, there are plenty of opportunities for OSS projects to use IT Infrastructure from other organizations without possessing them (e.g. GitHub as a DVCS).

What de Laat [1] describes as institutionalization in order to involve outside parties, is what the legal entity is in our framework.

In their paper, Lindman and Hammouda [48] illustrate the role of OSS foundations and the relationships between OSS communities and other OSS foundations. Although their unit of analysis was OSS foundations, the taxonomy can be compared to the organizational framework in our study; each of their elements can also be found in our organizational framework.

Our framework describes the different elements comprised within a single OSS project and shows the broad variety and complex constellation surrounding such a project. However, as an example, an OSS community may have different projects (e.g. different software products with different goals) and therefore a project does not correspond to the organization (the three dimensions of our framework). Moreover, an umbrella organization such as the Linux Foundation may offer their legal entity in order to protect the project and to offer services relating those three dimensions of our framework. Therefore, the legal aspect of the organizations does not need to correspond to the project. We see our framework as a first step to better understand and differentiate the concepts and different elements in order to ask more specific questions relating OSS research.

Our organizational framework broadens the view of Riehle and Berschneider [49] that shows 3 different ways in which a mature OSS project can govern itself in the future: 1) continue as is, 2) create its own legal entity or 3) affiliate with an existing OSS foundation. In our view, in addition to these 3 forms, an "in-between" solution involving collaboration with different OSS foundations is also possible. Such collaboration can be in all areas of our organizational framework or in specific areas only. This is the case in the LibreOffice community which has its own legal entity (The Document Foundation), but buys some services in the funding process from another foundation [50].

For practitioners, our framework will help to provide a better understanding of the structure an OSS project can have and how the different elements can be organized, similar to an organization. Moreover, our framework can provide practitioners valuable insights on several managerial aspects relating to OSS governance.

References

1. De Laat, P.B.: Governance of open source software: state of the art. Journal of Management & Governance. **11**, 165–177 (2007).
2. De Noni, I., Ganzaroli, A., Orsi, L.: The evolution of OSS governance: a dimensional comparative analysis. Scandinavian Journal of Management. **29**, 247–263 (2013).
3. Franck, E., Jungwirth, C.: Reconciling rent-seekers and donators–The governance structure of open source. Journal of Management and Governance. **7**, 401–421 (2003).
4. Schaarschmidt, M., Walsh, G., von Kortzfleisch, H.F.O.: How do firms influence open source software communities? A framework and empirical analysis of different governance modes. Information and Organization. **25**, 99–114 (2015).

5. Markus, M.L.: The governance of free/open source software projects: monolithic, multidimensional, or configurational? Journal of Management & Governance. **11**, 151–163 (2007).
6. Luhmann, N.: Funktionen und Folgen formaler Organisation. Berlin: Duncker & Humblot (1964).
7. Choudhury, V., Sabherwal, R.: Portfolios of Control in Outsourced Software Development Projects. Information Systems Research. **14**, 291–314 (2003).
8. Becker, J., Kahn, D.: The Process in Focus. In: Becker, J., Kugeler, M., and Rosemann, M. (eds.) Process Management: A Guide for the Design of Business Processes. pp. 1–12. Springer, Berlin, Heidelberg (2003).
9. O'Sullivan, A., Sheffrin, S.M.: Economics: Principles in Action. Upper Saddle River, New Jersey: Pearson Prentice Hall (2003).
10. IFRS Foundation: International Financial Reporting Standards, http://www.ifrs.org/issued-standards/list-of-standards/, Accessed 19.01.2018.
11. Rheingold, H.: The Virtual Community: Homesteading on the Electronic Frontier. Cambridge, Mass: MIT Press (2000).
12. von Krogh, G., Haefliger, S., Spaeth, S., Wallin, M.W.: Carrots and rainbows: Motivation and social practice in open source software development. MIS Quarterly. **36**, 649–676 (2012).
13. Deci, E.L., Ryan, R.M.: The general causality orientations scale: Self-determination in personality. Journal of Research in Personality. **19**, 109–134 (1985).
14. Roberson, Q.M., Stewart, M.M.: Understanding the motivational effects of procedural and informational justice in feedback processes. British Journal of Psychology. **97**, 281–298 (2006).
15. Raymond, E.S.: The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. O'Reilly Media, Inc., Sebastopol, CA, USA (2001).
16. German, D.M.: The GNOME project: a case study of open source, global software development. Software Process: Improvement and Practice. **8**, 201–215 (2003).
17. Bagozzi, R.P., Dholakia, U.M.: Open Source Software User Communities: A Study of Participation in Linux User Groups. Management Science. **52**, 1099–1115 (2006).
18. Iivari, N.: Empowering the users? A critical textual analysis of the role of users in open source software development. AI & SOCIETY. **23**, 511–528 (2009).
19. von Krogh, G., Spaeth, S., Lakhani, K.R.: Community, joining, and specialization in open source software innovation: a case study. Research Policy. **32**, 1217–1241 (2003).
20. Kilamo, T., Hammouda, I., Mikkonen, T., Aaltonen, T.: From proprietary to open source—Growing an open source ecosystem. Journal of Systems and Software. **85**, 1467–1478 (2012).
21. MacCormack, A., Rusnak, J., Baldwin, C.Y.: Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code. Management Science. **52**, 1015–1030 (2006).

22. Heppler, L., Eckert, R., Stuermer, M.: Who Cares About My Feature Request? In: IFIP International Conference on Open Source Systems: Integrating Communities. pp. 85–96. Springer, Cham (2016).
23. Anvik, J., Hiew, L., Murphy, G.C.: Who Should Fix This Bug? In: Proceedings of the 28th International Conference on Software Engineering. pp. 361–370. ACM, New York, NY, USA (2006).
24. Hars, A., Ou, S.: Working for free? Motivations of participating in open source projects. In: System Sciences, 2001. Proceedings of the 34th Annual Hawaii International Conference on. pp. 25-39. IEEE (2001).
25. O'Mahony, S., Bechky, B.A.: Boundary organizations: Enabling collaboration among unexpected allies. Administrative Science Quarterly. **53**, 422–459 (2008).
26. Bonaccorsi, A., Rossi, C.: Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. Knowledge, Technology & Policy. **18**, 40–64 (2006).
27. Lerner, J., Tirole, J.: Some Simple Economics of Open Source. The Journal of Industrial Economics. **50**, 197–234 (2002).
28. Berdou, E.: Insiders and outsiders: paid contributors and the dynamics of cooperation in community led F/OS projects. In: IFIP International Conference on Open Source Systems. pp. 201–208. Springer, Boston, MA (2006).
29. GNOME Foundation: GNOME Foundation - Fiscal Year 2016 Annual Report, https://www.gnome.org/wp-content/uploads/2017/07/GAR2016-web.pdf, Accessed 19.01.2018.
30. The Eclipse Foundation: Bylaws of Eclipse Foundation, Inc., https://www.eclipse.org/org/documents/Eclipse%20BYLAWS%202011_08_15%20Final.pdf, Accessed 19.01.2018.
31. Jensen, C., Scacchi, W.: Collaboration, leadership, control, and conflict negotiation and the netbeans.org open source software development community. In: System Sciences, 2005. p. 196b–196b. IEEE (2005).
32. Gamalielsson, J., Lundell, B.: Sustainability of Open Source software communities beyond a fork: How and why has the LibreOffice project evolved? Journal of Systems and Software. **89**, 128–145 (2014).
33. West, J., O'Mahony, S.: Contrasting community building in sponsored and community founded open source projects. In: System Sciences, 2005. HICSS'05. Proceedings of the 38th Annual Hawaii International Conference on System Sciences (2005).
34. Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., Ye, Y.: Evolution patterns of open-source software systems and communities. In: Proceedings of the International Workshop on Principles of Software Evolution. pp. 76–85. ACM (2002).
35. Dahlander, L., Magnusson, M.G.: Relationships between open source software companies and communities: Observations from Nordic firms. Research Policy. **34**, 481–493 (2005).
36. O'Mahony, S., Ferraro, F.: The emergence of governance in an open source community. Academy of Management Journal. **50**, 1079–1106 (2007).

37. Scacchi, W.: Understanding the requirements for developing open source software systems. IEE Software Proc. **149**, 24–39 (2002).
38. The Eclipse Foundation: Eclipse Development Process, https://www.eclipse.org/projects/dev_process/development_process.php, Accessed 19.01.2018.
39. O'Mahony, S.: Guarding the commons: how community managed software projects protect their work. Research Policy. **32**, 1179–1198 (2003).
40. Gutwin, C., Penner, R., Schneider, K.: Group awareness in distributed software development. In: Proceedings of the 2004 ACM conference on Computer supported cooperative work. pp. 72–81 (2004).
41. Kalliamvakou, E., Damian, D., Blincoe, K., Singer, L., German, D.M.: Open source-style collaborative development practices in commercial projects using github. In: Proceedings of the 37th International Conference on Software Engineering. pp. 574–585. IEEE Press (2015).
42. Ducheneaut, N.: Socialization in an Open Source Software Community: A Socio-Technical Analysis. Computer Supported Cooperative Work (CSCW). **14**, 323–368 (2005).
43. Jørgensen, N.: Putting it all in the trunk: Incremental software development in the FreeBSD open source project. Information Systems Journal. **11**, 321–336 (2001).
44. García-García, J., Alonso de Magdaleno, M.I.: Valuation of Open Source Software: How do you put a Value on free?. Revista de Gestão, Finanças e Contabilidade. 3 (2013).
45. Fitzgerald, B.: The transformation of open source software. MIS Quarterly. **30**, 587–598 (2006).
46. Anderson, H., Dare, T.: Passport Without A Visa: Open Source Software Licensing and Trademarks. International Free and Open Source Software Law Review. **1**, (2010).
47. Riehle, D.: The Economic Case for Open Source Foundations. Computer. **43**, 86–90 (2010).
48. Lindman, J., Hammouda, I.: Investigating Relationships Between FLOSS Foundations and FLOSS Projects. In: IFIP International Conference on Open Source Systems: Towards Robust Practices. pp.14-22. Springer, Cham (2017).
49. Riehle, D., Berschneider, S.: A Model of Open Source Developer Foundations. In: IFIP International Conference on Open Source Systems: Long-Term Sustainability. pp. 15–28. Springer, Berlin, Heidelberg (2012).
50. Software in the Public Interest, Inc., http://www.spi-inc.org/projects/libreoffice, Accessed 19.01.2019.