Exposing Differences of Governance Approaches in Single and Multi Vendor Open Source Software Development

Mario Schaarschmidt, Matthias Bertram, and Harald F.O. von Kortzfleisch

Universitätsstr. 1, 56070 Koblenz, Germany
Tel.: +49 (0) 261 287 2864, +49 (0) 261 287 2544, +49 (0) 261 287 2523
{mario.schaarschmidt,matthias.bertram,
harald.von.kortzfleisch}@uni-koblenz.de

Abstract. Research confirms that commercial OSS exists in many different ways according to its revenue model, type of license, development style, number of participating firms, number of participating volunteers or governance mode. In order to differentiate between an increasing variety of commercialization approaches, one may distinguish between projects with one dominating company, so called *single vendor projects* and those where more than one company is active, so called *multi vendor projects*. Furthermore, in order to structure different approaches, a project's history is equally of importance in terms of whether a project was initiated by a firm or a community. In this paper, we therefore analyze and compare single and multi vendor as well as *firm initiated* and *community initiated* OSS projects with regard to technical contribution of voluntary and paid project members. Based on a dataset build upon Eclipse projects we expose, that the number of paid members is significantly higher in firm initiated and multi vendor projects.

Keywords: Open Source Software, Single Vendor Projects, Multi Vendor Projects, Communities, Governance.

1 Introduction

The commercial production of open source software (OSS) has attracted a lot of attention in recent years (Dahlander and Magnusson 2005; Fosfuri et al. 2008). Success stories like Linux, MySQL or JBoss have proven that nowadays OSS has the quality and the customer acceptance to compete with its proprietary rivals. However, although the term open source suggests that software which claims to be OSS share a coherent body of attributes, at its core, the only connecting attributes are (1) delivery of the source code in a human readable form and (2) a license approved by the Open Source Initiative (OSI). Based on these two factors, many development styles or commercialization approaches are possible, which, although very different in terms of motivation and goals, are considered to be open source (Raymond 1999). For example, even Microsoft, a candidate for high quality proprietary software products has released open source licenses, such as Microsoft Reciprocal License (Ms-RL), which are consistent with OSI requirements.

Recent research confirms that commercial OSS exists in many different ways according to its revenue model, type of license, development style, number of participating firms, number of participating volunteers or governance mode (Bonaccorsi et al. 2006; Dahlander and Magnusson 2008; West 2003). Consequently, core business functions like community management, sales, marketing, product management, engineering and support differ among different commercialization strategies (Watson et al. 2008). For example, relevant revenue models range from dual licensing approaches, where a product is offered under two licenses, one OSS license and (at least) one proprietary license, to approaches where the revenue stream entirely is generated through the sale of complementary products or services (c.f. Alexy 2009; Fitzgerald 2008).

In order to differentiate between an increasing variety of commercialization approaches, one may distinguish between projects with one dominating company, so called *single vendor projects* and those where more than one company is active, so called *multi vendor projects*. Whereas single vendor approaches show similarities to proprietary software vendors' behavior (Riehle 2011), in cases of multiple firms active in a project, development is being processed like in R&D alliances or joint ventures (Schaarschmidt and Von Kortzfleisch 2009). In the latter case, usually a direct revenue stream is not intended. Instead, multiple firms combine their resources in order to build a platform and to promote standards with the aim to sell on top applications along with complementary products or services.

Furthermore, as shown by Dahlander (2007), in order to structure different approaches, a project's history is equally of importance in terms of whether a project was initiated by a firm or a community. However, despite the fact that there are differences among different approaches, yet, little is known about the differences in detail, e.g. in terms of governance or control. For example, what does it mean if a project is controlled by more than one firm? How does firm involvement or a project's history affect voluntary participation? In this paper, we therefore analyze and compare single and multi vendor OSS projects as well as firm initiated and community initiated ones with regard to technical contribution of voluntary and paid individual contributors. Based on a dataset build upon Eclipse projects we show, along with other results, that the number of paid contributors is significantly higher in firm initiated and multi vendor projects, reflecting a firm's wish to influence a project's trajectory.

2 Conceptual Background and Driving Phenomena

2.1 Commercial Open Source Software

OSS has come a long way. In the beginning, the majority of projects named OSS were initiated and driven by a handful of pioneers working for free, mostly due to fun in programming and problem solving or in order to build applications which were not available in a market (Bitzer et al. 2007; Shah 2006). Over the last decade, however, prices for proprietary licenses decreased resulting in an increased interest of adopting firms and commercial vendors in OSS. With the presence of firms in OSS projects, clear distinctions between proprietary and OSS products began to dissolve. Within the group of OSS projects Riehle (2009; 2011) distinguishes between community and commercial OSS. In his arguments, control and ownership structures are the critical indices in order to differentiate between these two types. While community OSS is controlled by a community of stakeholders (including multiple individual programmers and/or firms), commercial OSS is controlled by exactly one stakeholder with the purpose of commercially exploiting it. In addition, Dahlander (2007) focuses on a longitudinal perspective, including the history of a project as a means to categorize different approaches. By examining more than 60 successful open source projects in detail he formulated a 2x2 matrix indicating whether a project was started by a community or a firm or is driven by a community or firm, respectively.

Despite their ability to structure different types of OSS projects, both views share the limitation of incompleteness. While Riehle (2009) ignores the history of a project and (implicitly) assumes that control and ownership structures are stable and will not change over time, Dahlander (2007) subsumes both, single vendor and multi vendor projects under the same umbrella, namely, firm driven projects. To capture both limitations we propose a framework based on the distinction between the single and multi vendor projects (X-axis) and a project's initiation (Y-axis) (figure 1).

Table 1. A typology of commercialization approaches combining on Dahlander (2007) and Riehle (2009)

	Single vendor project	Multi vendor project
Firm initiated	Approach I	Approach II
Community initiated	Approach III	Approach IV

By looking at representative projects for each of the four approaches we find differences. Trolltech or MySQL are prominent examples for single vendor approaches and are characterized by the fact that one firm is the sole owner of the product they generate revenue from (Fitzgerald 2008; Watson et al. 2008). In cases of sole ownership of the entire code, dual licensing approaches are possible, meaning that a customer may chose between an OSS license without paying license fees and a more sophisticated version under a proprietary license the customer has to pay for. Although it is not entirely clear if these approaches are profitable – the most of these firms are not traded in a stock market and therefore do not have the obligation to publish their revenue figures – recent venture capital investment in OSS mirrors its potential (Schaarschmidt and Von Kortzfleisch 2010). According to a recent Gartner report, by 2012 more than 50% of all revenue generated from open source projects will come from projects under a single vendor's patronage (Riehle 2009). In contrast, in cases where many firms are active, like Linux or Apache, the code is not owned by a single firm which makes dual licensing approaches impossible. Moreover, those multi vendor projects usually aim to reducing costs for a product, each firm otherwise had to build alone and not to a direct revenue stream from license fees (in case of dual licensing) or complementary services.

Despite the presence of firms in the development of OSS, many projects rely on a heavy voluntary user and/or developer base. Even in single vendor approaches, where one might assume that marketing reasons are the predominant driver for offering OSS licenses, working with a community of users and developers is important for two reasons. Firstly, the risk of getting an evil reputation as a consequence of community mismanagement is too high. Secondly, and more importantly, these communities are valuable resources for OSS companies, as complementary assets in some cases and as will-be future employees in others (Dahlander and Wallin 2006). According to interviews with its CEO, JBoss, for example, recruits its future employees almost entirely out of the community of programmers. As a consequence, the extent of JBoss's contribution to their affiliated projects is between 60 and 95 % – although they do *not* own the code (Watson et al. 2008).

The presence or absence of a community of programmers external to a firm affects the way a project is managed. Depending on factors like type of license and business model, furthermore, the importance of a user and/or developer community varies for a firm active in an OSS project. Consequently, the importance and the extant of community management differ as well. In the following, we will formulate hypothesis to predict number of technical contributions as well as number of paid or unpaid individuals in single and multi vendor projects and, additionally, predict if the distribution of developers is dependent on a project's history.

2.2 Hypotheses Development

By considering number of firms participating in OSS projects, our framework implicitly puts emphasis on different governance modes. When talking about governance in the context of various theories of the firm, human resources a firm has to pay for to obtain differ from freely available resources in the following way. If someone is paid, generally speaking, in order to receive monetary compensation for the work which is pursued, he or she accepts certain responsibilities such as following a supervisor's authority. In contrast, at first glance, in cases of free contributors, those people are not legally bounded and therefore free to deicide to retire from a project at any time (Shah 2006). However, as several studies have revealed, giving signals to a future employee, status in a community of developers or fun in problem solving are motives for free programmers to stay with a project (Lerner and Tirole 2002; Bitzer and Geishecker 2010; Franck and Jungwirth 2003).

Firms applying a single vendor approach differ from firms active in a multi vendor project. As discussed in the section above, a community of voluntary developers is a valuable resource pool and complements a firm's own capabilities (De Laat 2007). In a similar vein, this holds true for firms cooperating with other firms in multi vendor projects. However, in order to complement own capabilities, the focal firm now gets access to other firm's resources. Furthermore, by sharing technical and financial risks with other firms, the importance of a developer community consisting of a high number of volunteers is decreasing.

An important part of a governance approach's composition is the role of leadership.[1] We therefore draw on the notion of leadership as a strong vehicle to obtain control (Jago 1982). Referring to our research objective we find formal and informal leadership structures such as those derived from following an archon and leadership structures as a consequence of contractual obligations. For example, based on interviews with Free/Libre Open Source Software development team members Scozzi and colleagues (Scozzi et al. 2008) suggested that leadership seems to be correlated with sustained contribution in these teams, pointing to a more informal leadership structure.

Drawing on different leadership types we therefore hypothesize, how the number of technical contributions, the number of committers, and the number of paid as well as voluntary project leaders differ between single and multi vendor projects. In this context committers are defined as programmers who are allowed to change parts of the projects source code.

Hypothesis 1:	*Multi vendor projects receive more technical contributions by firms than single vendor projects.*
Hypothesis 2:	*The number of paid committers is higher in multi vendor projects.*
Hypothesis 3:	*The number of voluntary project leaders is higher in single vendor projects.*
Hypothesis 4:	*The number of paid project leaders is higher in multi vendor projects.*

Dahlander (2007) further points to the importance of a project's history for different governance modes. The evolution of many OSS projects has shown that a founder's personality as well as his technical abilities influences the composition and activity of a community (O'Mahony 2007). Linux, one of the big success stories within the OSS movement, until now is dependent on Linus Torvalds, its founder. We therefore propose that a project initiated by a community of developers (or a sole influencing developer) differs from a project initiated by a firm. For instance, to avoid the impression of harnessing a community's work, firms entering a stable community initiated project are likely to limit their effort in managing and controlling a project's trajectory (West and O'Mahony 2008). Therefore, we assume community initiated projects to consist of less paid project leaders than their firm initiated counterparts. In addition, due to a founder's personality, community initiated projects might be more attractive for other voluntary programmers (Stewart and Gosain 2006). In contrast, projects initiated by one or more firms are most likely to be led by paid project leaders in order to align with a firm's goal.

Hypothesis 5:	*Community initiated projects receive more technical contributions by volunteers than firm initiated projects.*
Hypothesis 6:	*The number of paid committers is higher in firm initiated projects.*
Hypothesis 7:	*The number of voluntary project leaders is higher in community initiated projects.*
Hypothesis 8:	*The number of paid project leaders is higher in firm initiated projects.*

[1] We skip a discussion about transaction cost economics here and point to the seminal work of Oliver Williamson.

3 Methodology

3.1 Research Setting: The Case of the Eclipse Foundation

In a search for data reflecting our research question, namely to identify different governance approaches for multi and single vendor OSS projects, we found the Eclipse foundation a suitable case to study. Eclipse itself is a hybrid, not just being a project; it is also a foundation which hosts several other projects. The foundation is one of the most successful ones with more than 100 members beside Mozilla and Apache Foundation. Characteristic for Eclipse is that a number of governance mechanisms are publicly available like the process of becoming a contributor or the responsibilities of its members. Furthermore, in the case of Eclipse, governance rules ignore the size of a firm. Every strategic board member has only one vote even if they donate much more than the others. The foundation's website provides a lot of information concerning Eclipse projects like the name of every committer, his affiliation, the status of a project, and especially the commitments to a project.

Eclipse as both, software product and foundation has a fascinating history. Eclipse was a development environment originated inside the boundaries of IBM. The major competitors to the Eclipse development environment were Microsoft's Visual Studio and Sun's NetBeans. To gain momentum IBM open sourced its development although they were sharing a $ 40 million dollar investment with its competitors (Wagstrom 2009). However, other vendors were now able to build their products on top of Eclipse rather than using proprietary software from competitors. Although in general every individual is welcome, possible future committers have to run through a process where they have to prove their programming qualification. Additionally, voluntary contributors as well as participating firms have to agree to certain process rules and a project charter. Furthermore, it is worth noting that every project is based on the principle of meritocracy which means that the more you contribute, and the higher the quality of your contribution, the more you are allowed to do.

3.2 Research Approach

We took data from the Eclipse website both manually and using a web miner, and stored it in a relational database. The data served as a pool to identify a number of variables we discuss in the following.

Technical Contribution

OSS vendors often use software tools like the Concurrent Versioning System (CVS)[2], Subversion (SVN)[3] or Git[4] to organize their source code in a so called software repository. In simple terms the software repository can be described as a container keeping all technical information (source code, Texts, images, database scripts…) necessary to build a software product. Furthermore through dedicated check-in/out and versioning functionalities the software repository enables developers to keep

[2] http://www.cvshome.org/
[3] http://subversion.apache.org/
[4] http://git-scm.com/

track of contributions to a software repository. This becomes important throughout the development and build phase of a software project. During the development phase a developer need to constantly add new, change existing or delete obsolete source code from the software repository.

Each contribution to a software repository as described above is logged by the software repository. By that, researchers may not only analyze the source code at the actual point in time but also gather information about the contributions made to the source code over a whole period of time. However, a recorded commit ignores the size of an actual contribution (Arafat and Riehle 2009). Therefore, it is not possible to identify directly if a contributor changed a whole function or just a few lines. Despite this limitation, many commits signal firm or community activity and a party's interest in a project - regardless of the actual size of the commit. We therefore coded technical contribution as the natural log of a commit made by a committer either being a volunteer or employed by a firm. Only projects where at least one commit occurred in the observed period are considered (December 2006 till December 2008).

Number of Committers
We were able to isolate information about each programmer's affiliation as well as his role within a project (committers can be active in more than just on project). Only those who have received the status of a *committer* are allowed to change parts of the source code. As the right to change is important in order to influence a project's trajectory, we only considered programmers with committers status – ignoring other types of contributions like bug identification or change requests. Based on our dataset, we were further able to distinguish between those paid by a firm and those who contributed voluntarily.

Number of Project Leaders
In a similar vein, the number of project leaders was counted. Again, as the Eclipse website provides a clear affiliation of each programmer as well as his role, leaders with a firm affiliation are coded as paid leaders; programmers without any affiliation are coded as voluntary leaders.

Project Initiation
In order to identify if a project is initiated by a firm or a community, we checked for who submitted the first commit for the focal project. In few cases first commits came from developers affiliated with a firm and voluntary developers. We then distinguished based on the number of commits. For example, if a project was founded in August 2004 and received 500 commits by firms and 20 by volunteers in the first month, this project was coded to be a firm initiated project.

Project Vendor Type
With our dataset we were also able to count for the number of firms in projects. As the Eclipse foundation has a commercial focus, we were not able to find a single project without firm participation. Based on the number of firms, we coded single vendor projects those who received commits by only one entity. Consequently, in cases of two or more firms active in project this project was considered a multi vendor project.

3.3 Research Results

After checking for projects with little or no activity or one without any commits, we ended up having a list of 83 projects. Of these projects, 17 were single vendor projects, 66 multi vendor projects, 33 were community initiated, 50 were firm initiated. In order to illustrate commit different behavior in single and multi vendor projects we chose two representatives (figure 1).

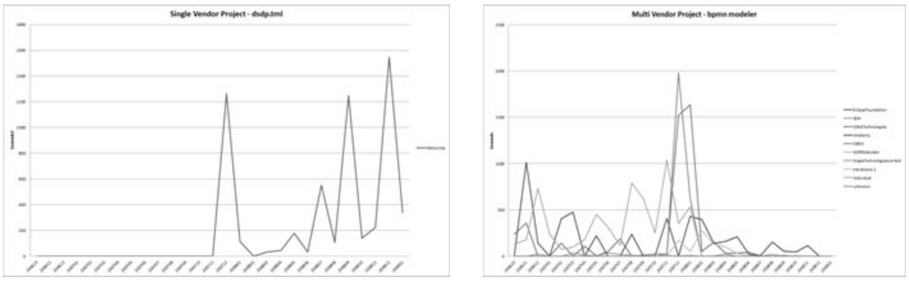

Fig. 1. A comparison between selected single (left) and multi (right) vendor projects

However, to test our hypotheses, we conducted an analysis of variance (ANOVA) to compare the mean values for each of our categorical variables. Specifically, we examined whether the mean values for the numbers of commits of firms, of volunteers, and number of paid project leaders in a project differ between different approaches. Although of high interest, we did not check for any combinations, such as community initiated single vendor projects. We abandoned this investigation due the impossibility to run statistical methods on small data sets. To run the ANOVA, we split the sample of projects into single vendor and multi vendor projects for a first analysis (table 1) and into firm initiated and community initiated for a second one (table 2).

Consistent with H1, the results show that multi vendor projects receive significantly more commits than single vendor projects. Furthermore, it is worth noting, that with regard to H2 the number of paid committers in multi vendor projects is nearly three times the number of committers in single vendor projects. As expected, the distribution of voluntary and paid project leaders among single and multi vendor projects reflects our hypotheses H3 and H4. However, it is worth noting that the overall number of voluntary project leaders is indeed very low (mean of 0.35 in single vendor, 0.12 in multi vendor projects). To further illustrate differences between single and multi vendor projects we calculated the ration of paid to all committers. Here we find multi vendor projects to have 87.54 % paid committers in contrast to 76.06 % in single vendor projects.

In a line with H5, community initiated projects receive significantly more commits by volunteers than their firm initiated counterparts. Interestingly, the number of commits by firms does not differ between community and firm initiated projects. We also find support for H6 in that the number of paid committers in firm initiated

Table 2. Means (M) and standard deviations (SD) after ANOVA on factor VENDORTYPE

	Single vendor project		Multi vendor project	
	M	SD	M	SD
Number of commits by firms	7.41[a]	*n.c.*	9.06[b]	*n.c.*
Number of commits by volunteers	4.08[a]	*n.c.*	5.05[a]	*n.c.*
Number of all commits	8.63[a]	*n.c.*	9.37[a]	*n.c.*
Number of voluntary committers	2.12[a]	3.57	1.80[a]	2.63
Number of paid committers	5.18[a]	2.90	14.83[b]	15.92
Number of voluntary project leaders	0.35[a]	0.61	0.12[c]	0.45
Number of paid project leaders	0.88[a]	0.60	1.58[b]	1.16
Ratio paid to all committers	76.06[a]	35.13	87.54[b]	17.14

Notes: Mean values reported, N=82. Within each row, means with a *different* superscript are significantly ($p < .05$ for b; $p < .1$ for c) different from each other. *n.c.*=not calculated by SPSS

Table 3. Means (M) and standard deviations (SD) after ANOVA on factor INITIATION

	Community initiated		Firm initiated	
	M	SD	M	SD
Number of commits by firms	8.10[a]	n.c.	8.95[a]	n.c.
Number of commits by volunteers	6.07[a]	n.c.	4.03[b]	n.c.
Number of all commits	9.26[a]	n.c.	9.11[a]	n.c.
Number of voluntary committers	2.36[a]	3.53	1.60[a]	2.26
Number of paid committers	8.45[a]	6.44	15.46[b]	17.77
Number of voluntary project leaders	0.21[a]	0.48	0.16[a]	0.51
Number of paid project leaders	1.33[a]	1.13	1.48[a]	1.09

Notes: Mean values reported, N=83. Within each row, means with the *same* superscript are not significantly ($p < .05$) different from each other. *n.c.*=not calculated by SPSS

projects is nearly twice the number of paid committers in community initiated ones. With regard to H7 and H8, namely the leadership structure in both types of projects, we do find figures reflecting our arguments; however, they are not significant. As we expected the number of paid project leaders to be significantly higher in firm initiated

Table 4. Overview on supported and not supported hypotheses

Hypothesis	Considered variable	Is significantly higher in	*Supported*
H1	Number of commits by firms	Multi vendor projects	Yes
H2	Number of paid committers	Multi vendor projects	Yes
H3	Number of voluntary project leaders	Single vendor projects	Yes
H4	Number of paid project leaders	Multi vendor projects	Yes
Hypothesis	Considered variable	Is significantly higher in	*Supported*
H5	Number of commits by volunteers	Community initiated	Yes
H6	Number of paid committers	Firm initiated	Yes
H7	Number of voluntary project leaders	Community initiated	No
H8	Number of paid project leaders	Firm initiated projects	No

Notes: A hypothesis is considered to be significantly supported if $p<.05$

projects, especially the comparatively high number of paid project leaders in community initiated projects was a surprise. Putting single and multi vendor projects together might be a threat in order to distinguish between community and firm initiated projects (see beginning of section *Research Results* for reasons to combine both approaches). Table 4 gives an overview on supported and not supported hypotheses.

4 Discussion and Conclusion

The aim of the paper was to find differences in governance approaches between single and multi vendor projects as well as between community and firm initiated ones. In our study we therefore focus on areas somewhat neglected by dominant OSS research streams. The objective of our research was a number of projects hosted under the umbrella of the Eclipse foundation. We treated leadership as a strong vehicle for firms to obtain control in an OSS projects. However, in a similar vein, having a high number of paid committers who have the right to change parts of the software code is equally an important instrument to perform control. We find support for six of our eight major hypotheses. Before discussing our results in the light of existing and future research, we have to point to a number of limitations.

Firstly, one important limitation is that we were not able to identify a measure for a project's size. Yet especially the size of a project, for instance, measurable in lines of code, determines the number of contributors and committers in a project. Secondly, as stated in the research results section, we were not able to run an ANOVA for each quadrant due to the limited size of our data set. In order to further distinguish approaches within the group of multi vendor projects, looking to a project's history (community vs. firm initiation) is an avenue for further research. Finally, we have to be careful in generalizing our results. Projects hosted by the Eclipse foundation inherently have a commercial focus, which means that conclusions from this research cannot be easily extrapolated to other OSS cases. Furthermore, the Eclipse foundation oftentimes acts as a platform for selling complementary products. For instance, Deutsche Post, a major German logistic company, has developed a service-oriented IT infrastructure over the last eight years. Service-oriented architectures (SOA) are a heavily discussed topic in IT settings due to the ability to create a flexible IT infrastructure. Recent developments at Deutsche Post made it necessary to offer the in-house solution, a SOA-framework, under the umbrella of the Eclipse Foundation as an openly available product named Swordfish; hoping for external contributions to the ongoing process of maintaining the software. However, although being a single vendor project, the product was never intended to be sold to third parties like in cases of dual licensing. Moreover, several firms use Swordfish as a basis for on-top products and solutions. The fact that the Swordfish project has failed to attract an active developer community external to the firm until now, points to challenge for single vendor, firm initiated projects to establish sustainable external contributions.

Drawing on our hypothesis we find different approaches to administrate commercial OSS projects from a firm's point of view which we will discuss in detail. Single vendor approaches are characterized by the fact that only one firm is active in an OSS project. The majority of these firms are startups or SMEs which need a community of developers to complement missing own resources. For them, having programmers external to the firm they do not have to pay for is crucial to compete with sustained rivals. In addition, when a firm wants to internalize resources, the developer community is a talent base they can directly recruit from. This, on the other hand, signals to voluntary programmers who then might spend their time for a project which they can benefit from in the long run.

In contrast, multi vendor projects are characterized by more than one firm. From a firm's point of view, as other firms also spend resources or pay developers to work in the project, the need to complement missing internal resources through volunteers decreases. Multi vendor projects therefore permit firms to complement resources not to be found internally with other firm's resources. These resources are usually skillful paid developers. Although they are still external to the focal firm, they are legally bounded to the project through contracts with external firms. However, as a downside, multi vendor projects inherent the problem of multiple firm interests which results in a high number of paid project leaders. As a consequence, these projects seem to be less attractive for voluntary programmers due to an increased firm presence.

Community initiated projects receive more commits by volunteers and consist of more voluntary project leaders than firm initiated projects. These findings mirror the assumption that individual programmers follow a founder or a group of founders, respectively. For firms entering such projects, in order to not disrupt established informal structures and norms they have to accept that too much influence, e.g. by sending too many paid programmers to work for the project, inherent the risk of a fork as a consequence of losing contact to community rules.

Firms initiating an OSS project have to put emphasis on the development of both, a user and a developer community. Depending on the type of software various strategies are possible. In cases of technology exploitation, where a firm opens an internally developed product it is important to know, if the firm wants to capture value itself or if it accepts to share value with others. In the latter case, the community of developers consists of programmers of other firms which also have an interest in the open project. In cases where a firm, e.g. a startup, offers a product under an OSS license, usually first efforts have to be made to establish a user community which then over time evolves to a community of voluntary developers.

As our study has shown, OSS projects differ among different dimensions and therefore require different management approaches. While sending paid programmers to established OSS communities can be useful in one case, this can be the wrong strategy in another. We therefore call for future research of differences in managing commercial OSS communities.

References

Alexy, O.: Free Revealing. How Firms Can Profit From Being Open. Gabler, Wiesbaden (2009)

Arafat, O., Riehle, D.: The Commit Size Distribution of Open Source Software. In: Proceedings of the 42nd Hawaiian International Conference on System Science (HICSS-42) (2009)

Bitzer, J., Schrettl, W., Schröder, P.J.H.: Intrinsic Motivation in Open Source Software Development. Journal of Comparative Economics 35, 160–169 (2007)

Bitzer, J., Geishecker, I.: Who Contributes Voluntarily to OSS? An Investigation Among German IT Employees. Research Policy 39, 165–172 (2010)

Bonaccorsi, A., Giannangeli, S., Rossi, C.: Entry Strategies Under Competing Standards: Hybrid Business Models in the Open Source Software Industry. Management Science 52(7), 1085–1098 (2006)

Dahlander, L.: Penguin in a New Suit: A Tale of How De Novo Entrants Emerged to Harness Free and Open Source Software Communities. Industrial and Corporate Change 16(5), 913–943 (2007)

Dahlander, L., Magnusson, M.G.: Relationships Between Open Source Software Companies and Communities: Observations From Nordic Firms. Research Policy 34(4), 481–493 (2005)

Dahlander, L., Magnusson, M.G.: How Do Make Firms Make Use of Open Source Communities? Long Range Planning 41, 629–649 (2008)

Dahlander, L., Wallin, M.: A Man on the Inside: Unlocking Communities as Complementary Assets. Research Policy 35, 1243–1259 (2006)

De Laat, P.B.: Governance of Open Source Software: State of the Art. Journal of Management and Governance 11(2), 165–177 (2007)

Fitzgerald, B.: The Transformation of Open Source Software. MIS Quarterly 30(3), 587–598 (2006)

Fosfuri, A., Giarratana, M., Luzzi, A.: The Penguin Has Entered the Building: The Commercialization of Open Source Software Products. Organization Science 19(2), 292–305 (2008)

Franck, E., Jungwirth, C.: Reconciling Rent-Seekers and Donators - The Governance Structure of Open Source. Journal of Management and Governance 7(4), 401–421 (2003)

Jago, A.G.: Leadership: Perspectives in Theory and Research. Management Science 28(3), 315–336 (1982)

Lerner, J., Tirole, J.: Some Simple Economics of Open Source. Journal of Industrial Economics 50(2), 197–234 (2002)

O’Mahony, S.: The Governance of Open Source Initiatives: What Does It Mean to Be Community Managed? Journal of Management and Governance 11(2), 139–150 (2007)

Raymond, E.S.: The Cathedral and the Bazaar (1999), `http://www.catb.org/~esr/writings/cathedral-bazaar/` (last access: 10/21/2010)

Riehle, D.: The Commercial Open Source Business Model. In: Proceedings of the 15th American Conference on Information Systems (AMCIS), San Francisco, CA (August 6-8, 2009)

Riehle, D.: The Single Vendor Commercial Open Source Business Model. Information Systems and e-Business Management (2011) (forthcoming)

Schaarschmidt, M., Von Kortzfleisch, H.: Divide et Impera! The Role of Firms in Large Open Source Software Consortia. In: Proceedings of the 15th American Conference on Information Systems (AMCIS), San Francisco, CA (August 6-8, 2009)

Schaarschmidt, M., Von Kortzfleisch, H.: The Business of Venture Capital in Open Source Software. Working Paper, presented at 10th EURAM Conference, Rome, Italy (May 19-22, 2010)

Scozzi, B., Crowston, K., Eseryel, Y., Li, Q.: Shared Mental Models Among Open Source Software Developers. In: Proceedings of the 41st Hawaii International Conference on System Science, HICSS-41 (2008)

Shah, S.: Motivation, Governance, and the Viability of Hybrid Forms in Open Source Development. Management Science 52(7), 1000–1014 (2006)

Stewart, K.J., Gosain, S.: The Impact of Ideology on Effectiveness in Open Source Software Teams. MIS Quarterly 30(2), 291–314 (2006)

Wagstrom, P.A.: Vertical Interaction in Open Source Software Engineering Communities, PhD Thesis, Carnegie Mellon University, Pittsburgh, PA (2009)

Watson, R.T., Boudreau, M.-C., York, P.T., Greiner, M., Wynn, D.: The Business of Open Source. Communications of the ACM 51(4), 41–46 (2008)

West, J.: How Open is Open Enough? Melding Proprietary and Open Source Platform Strategies. Research Policy 32, 1259–1285 (2003)

West, J., O’Mahony, S.: The Role of Participation Architecture in Growing Sponsored Open Source Communities. Industry and Innovation 15(2), 145–168 (2008)