Firms on SourceForge

Eilhard, Jan
Cerna, École Nationale Supérieure des Mines de Paris

28. January 2008

Firms on SourceForge - Loose Contracts, Tight Control?

Corporate Contribution and Leadership in SourceForge Projects

January 28, 2008

Jan Eilhard

Cerna

Ecole Nationale Superiéure des Mines de Paris, France

Abstract

This paper explores empirically what factors influence a firm's decision to contribute and to take leadership in open source projects. Increasing firms' participation in the development of open source software (OSS) is generally perceived as a puzzle. Assuming that firms face a "Make-or-Buy" decision before using OSS, we argue that contribution is in fact the best way for them to keep control of their supplier in a context where incomplete open source licenses govern transactions. Building on this proposition, we derive predictions on the drivers of firms' contribution and leadership in open source projects, and test them on a unique dataset of 4,808 open source projects extracted from Sourceforge. Our empirical findings confirm the predictions and lend support to our hypotheses.

Acknowledgements

This working paper was prepared for the chair of Law and Economics of Patents at Cerna, Ecole Nationale Superiéure des Mines de Paris. It is part of the EU Patent Reform program. Financial support by Air Liquide, Alcatel, Microsoft, Philips, SAP is gratefully appreciated. Professeur François Lévêque and Yann Ménière provided invaluable support and comments.

For questions and suggestions:
Jan Eilhard
Cerna, Ecole Nationale Superiéure des Mines de Paris
60, Boulevard Saint Michel
75272 Paris Cedex 06
France
email address: eilhard@ensmp.fr

Introduction

The use of open source software (OSS) is quickly becoming commonplace in enterprises. Ranging from entire software platforms- Linux immediately comes to mind- to Internet tools, like sendmail or PHP, firms implement a wide array of open source solutions. But not only is the available software manifold, the firms that use open source are diverse, too. Among them are large, multinational hardware producers such as IBM, Sun and Nokia as well as small, medium-sized software vendors. Although commercial open source use stirs the attention of industry experts and the media, for economists, it does not pose too big a puzzle. As soon as OSS performs a task well enough, it is a simple substitute for proprietary programs, coming free of charge. Within their supply chain, companies replace proprietary software with open source alternatives as complements to their products or a technology input in their production. In fact, little does it surprise then that firms use OSS.

In contrast, one phenomenon *is* rather startling: Firms are strongly involved in OSS. According to a recent study, firms develop 15% of all open source software world-wide (UNU-MERIT, 2006). More strikingly, Lakhani & Wolf (2005) find that around 40% of open source developers are paid to contribute to projects. This raises an important question: Why do firms participate in open source projects? Open source licenses only grant few exclusive rights to the licensor. Unlike proprietary licenses, they allow anybody to download and run the software. Hence, since the source code of OSS is essentially non-excludable, companies have difficulties appropriating their contributions. They have to share the returns on their investment with the open source community. This very fact has spawned a burgeoning field of empirical research (Wichmann, 2002; Bonaccorsi & Rossi, 2003; Dahlander, 2005; Henkel, 2006). The issue of corporate control over open source projects however has received little attention as yet.

Control and governance are important aspects in transactions. A traditional transaction generally takes place with fixed product specifications, delivery dates, and fees in case of non-compliance. Contracts outline the conditions of a transaction and capture foreseeable circumstances. Under unforeseen circumstances however contracts are incomplete and transacting partners face uncertainty and the risk of opportunistic behavior *ex post*. Trading partners cope with this problem through *ex ante* arrangements that set the degree of vertical integration (Williamson, 1967). Control thus arises through contracts and the governance structure between trading partners.

This is not the case in open source development. Albeit firms and open source projects transact in terms of applications, new features or bug fixes, the contract, the open source license in this case, offers no means of coercion. A company cannot force an open source community to provide a particular feature, to fix a programming bug or to develop a project in the direction the firm choses. The license only sets a mutual framework for collaboration. There is however a way firms may influence open source projects: They contribute source code. By participating, firms affect the governance structure and can control their open source transactions.

Companies can tighten their control even further by taking project leadership. Projects on SourceForge (SF) have a clear hierarchy. Project administrators lead the development. They can contribute source code, grant developers access to the code base, and ultimately decide which contribution is accepted in a project. They effectively have control over the direction the project takes. Developers, on the other hand, only contribute code. They can neither decide which contribution is accepted nor which version of the project is released.

Our objective is to explore what factors influence a firm's decision to contribute and to take project leadership in SF projects. To this end, we relate data on SF projects with corporate participation and project leadership. Our paper is unique because we can track these two ways of firm involvement for SF projects. The paper proceeds as follows. We first present the related literature on corporate control and open source software. Then we discuss our methodology and the data, after which we interpret our results. Finally, we conclude our analysis and give an outlook on future research.

Corporate Control and Open Source

The choice of "Make-or-Buy" is at the core of corporate use of open source software. Although corporate contribution does not necessarily follow from use, it is a nontrivial element in the interaction between a company and an open source project. These interactions rely on rather incomplete contracts, namely the open source licenses. Transactions under incomplete contracts are prone to uncertainty and opportunistic behavior (Williamson, 1967). To control these transactions, especially if trading partners incur *ex ante* investments that are not, or barely, useful for other transactions, further governance arrangements are needed. These arrangements set the extent to which one trading partner integrates the other one (Grossman & Hart, 1986). The properties of the transaction are important factors for integration (Klein, 1998): the investment that is required *ex ante*, the degree of uncertainty about the transaction, the complexity of the interaction and its frequency.

Firms and open source communities transact in terms of project improvements, such as bug fixes or additional features, as well as services, for instance documentation and support. These transactions are based on incomplete contracts, the open source licenses, and hence require additional governance arrangements. Setting up these arrangements is costly however and so a firm attempts to increase the net value of outsourced technology per transaction (Veugelers & Cassiman, 1999). Software production involves strong economies of scale with large up-front fixed cost and small marginal cost. Hence large producers can create software at lower cost. Along these lines, we expect to find that

> Prediction 1: *...firms are more likely to contribute in large projects.* We use the number of downloads and project age as proxies for the project size.

The necessity to set up governance increases with the extent of incompleteness within the contract (Williamson, 1967). A contract that covers few contingencies leaves a trading partner prone to uncertainty and risk of opportunism. She establishes additional control measures to reduce this risk. An open source contract only provides for a minimal set of parameters. As a consequence, the more parameters a transaction between firm and open source community involves the stronger is the need to control through contribution. Accordingly, we predict that

> Prediction 2: *...firms are more likely to contribute in projects that are complex*. We use the number of multiple audiences, of programming languages, of topics and of feature requests as proxies for the complexity of the transaction and hence of the incompleteness of the contract.

Open source software can threaten a firm's competitive advantage (Henkel, 2006). The use of OSS can give a firm's rival access to essential information about a product or, in the worst case, the product itself. This issue is similar to the one firms face in R&D outsourcing (Love & Roper, 2002). Potential information leakages in external R&D endanger a firm's competitive stance and are an impediment to outsourcing. Therefore external R&D is more likely for generic technology inputs (Kamien & Zang, 2000). To protect one's competitive advantage, a firm outsources R&D efforts that center on process rather than product innovations. We predict therefore that

> Prediction 3: *...firms are more likely to contribute in projects that deal with processes rather than products.* This means that firms prefer projects that target more technically versed audiences, such as system administrators and developers, than end-users and product markets.

Appropriation is essential for technology outsourcing (Gooroochurn & Hanley, 2007). Common means of intellectual property protection are patents, copyrights, secrecy, lead time and complementary assets. All of which are also used in the interaction between companies and open source projects (Dahlander & Magnusson, 2005). Companies can for example focus on projects with Academic licenses (Raymond, 2001) allowing them not only to use the contributions from the open source community in proprietary software, but also to keep their own contribution secret. We contend therefore that

> Prediction 4: *...firms are more likely to contribute in projects with Academic licenses.* We divide open source licenses into the categories of Academic, Weak Copyleft and Strong Copyleft licenses.

Data

Our data base consists of projects posted on SourceForge. SourceForge (SF) is an Internet platform that acts as an intermediary between project initiators, developers and users of open source software. It is the largest platform of its kind and includes more than 1,000,000 subscribers and approximately 130,000 projects. Subscription to SF is free and only needed to participate in any phase of project development, i.e. for fixing a bug, contributing code or authorizing another release. SF offers the necessary infrastructure to maintain developer communities, the bandwidth for downloads as well as facilities to manage mailing lists and search facilities for users.

We focus our analysis on projects that are often downloaded and most actively developed. Choosing projects that are in the first quartiles of number of developers as well as of number of downloads, we reduce the dataset to 4,808 projects. Previous studies (Howison & Crowston, 2004; Lerner & Tirole, 2005; David & Rullani, 2006) show that SF contains a large number of small or inactive projects. Selecting larger projects may partly offset the negative bias within SF projects. Lastly, we convert the initial panel data into a cross-section. We average quantitative variables and use the modes for each category for qualitative ones.

The unit of observation is the software project. We collect quantitative as well as qualitative variables for each project. Quantitative data contains, among others, the number of developers, the number of downloads and the number of firms. Qualitative data consists of project characteristics such as which type of open source license the project uses, for which operating system a project is developed or for which audience the project is intended. We observe a time period spanning 21 months from February 2005 until October 2006.

The data comes from two databases: the University of Notre Dame and the FLOSSmole project. Both databases acquire their statistics from SF. However, the two use different methods. While the University of Notre Dame directly obtains monthly snap shoots of SF's database, FLOSSmole runs a web application that gathers data from projects' web pages. Due to possible technical problems with the websites, the information from the FLOSSmole project can be slightly less accurate. There is however a severe measurement error in the University of Notre Dame data. Numbers of downloads and developers are incorrect and we therefore have to rely on the FLOSSmole data for these two variables.

The email addresses of developers and project leaders are available from the University of Notre Dame database from December 2004 until October 2006. We match email addresses with user registration for projects and obtain 15,307 top-level domains. These domain entries are then inserted into an application that fetches the corresponding website and searches for keywords with respect to three different categories: *Academic*, *Corporate* and *Private*. The list of keywords was obtained through a preliminary search on ten websites for each respective category. We rank the domains according to the number of times keywords appear on the website. In the end, we thus obtain 8,007 valid domain and category pairs. In terms of developers, this means that we relate 20,872 developers with either one of the categories. From these, we can identify 5,233 corporate developers.

There are two main sources of measurement errors within the data. First, there is a sample selection bias in the SF data for small and medium-sized projects. Focusing on the largest projects in terms of downloads and developer subscriptions should alleviate this problem a little. Without a comparable dataset with projects outside a repository however, the result cannot be verified for statements on OSS in general.

Second, type I and type II errors on corporate affiliations are likely due the method we apply. Corporate developers might use their work email addresses for private purposes. Thus, although we find a corporate affiliation, there is in fact no coordinated firm strategy on OSS. The potential type II error goes the opposite way. Corporate developers could use their private email addresses for work related projects. Without interviewing each developer, there is no possibility of knowing how strong these two effects are. However, the results could suggest how well our affiliation indicator works. In case either the type I or II errors are strong, one would expect white noise in the analysis. Thus, by finding statistically significant results, one could argue that the two errors are relatively small.

Discussion

Descriptive Analysis. Figure 1 shows the distribution of licenses in our sample. Projects with a Strong Copyleft license are most common (67%). Academic licenses take up 16%, whereas 16% of the projects run under a Weak Copyleft license. The predominance of the Strong Copyleft licenses, especially the GPL license, among projects on SF is well-documented. This has historic reasons as the GPL is the oldest and best-known open source license and project initiators, when picking from a myriad of different licenses, choose the one that they know best. Figure 2 presents the frequency of software platforms for which the projects are mainly developed. Projects are almost equally often developed for POSIX operating systems (37%) as for middleware applications (34%). POSIX is a standard that is compatible with UNIX-like operating systems, such as Linux, BSD or Solaris. Middleware applications, on the other hand, are platform independent. They are written in programming languages that are instantaneously read, using third-party software to interact with the operating system. 23% of all projects are developed for Windows. The sample also contains projects written for the Mac OS X (3%) and other platforms (3%), for instance for portable devices.

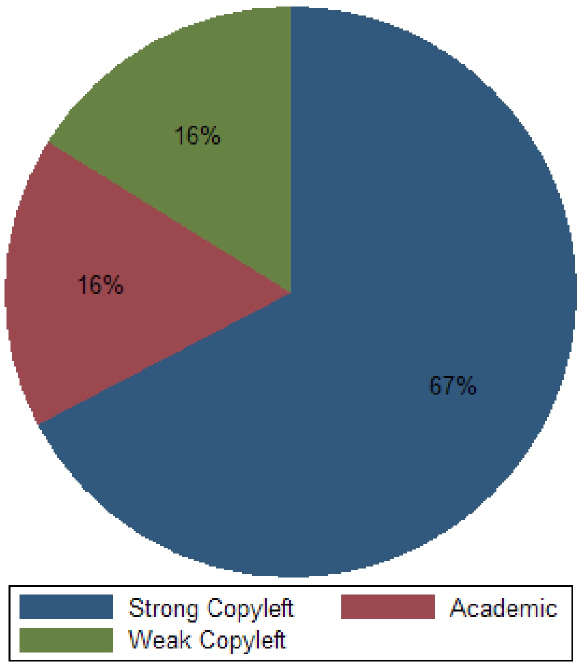


Figure 1: Proportion of Licenses

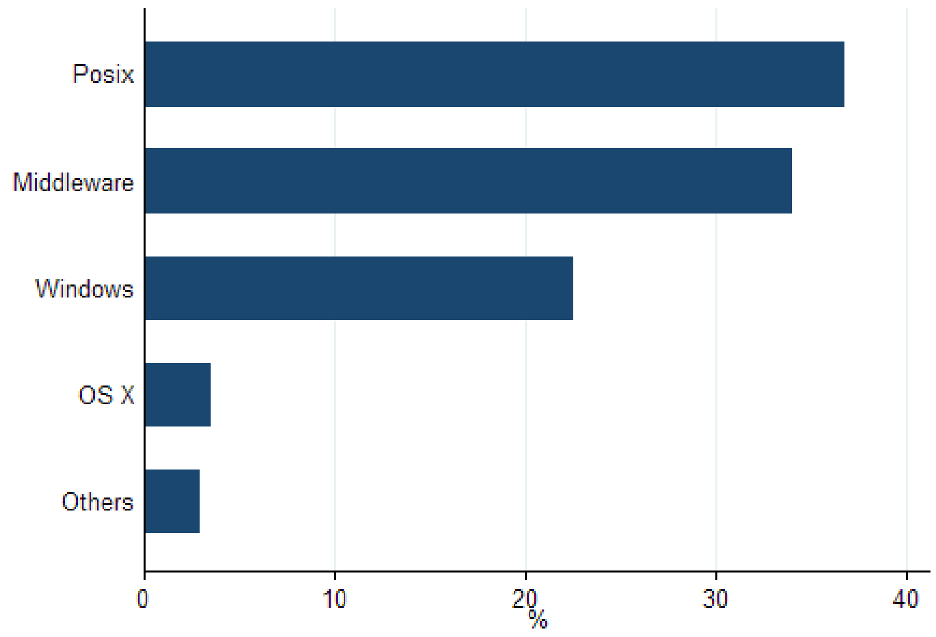


Figure 2: Proportion of Software Platforms

Table 1 presents summary statistics. The number of downloads per project is on average 14,116 (s.d.: 270,634). Our sample only includes projects that have existed for at least ten months (320 days) and on average around 3 years (1,232 days). The average numbers of corporate developers and project leaders are 1.09 (s.d.: 2.75) and 0.49 (s.d.: 0.93) respectively. The number of total developers per project, 7.12 (s.d.: 4.79), is rather high compared to similar research (Bonaccorsi & Rossi, 2003).

Table 1: Descriptive Analysis

Variable	Obs	Mean	Std. Dev.	Min	Max
Downloads	4808	14,116	270,634	10	16,000,000
Project Age (Days)	4808	1,232	492	320	2,190
Academic Developers	4808	1	3.26	0	122.75
Corporate Developers	4808	1	2.75	0	86.83
Private Developers	4808	2	4.38	0	113.17
Other Developers	4808	0.07	0.44	0	12.83
Academic Project Leaders	4808	0.33	0.87	0	12.75
Corporate Project Leaders	4808	0.49	0.93	0	12.92
Private Project Leaders	4808	1	1.15	0	12.25
Other Project Leaders	4808	0.04	0.23	0	4
Developers	4808	7.12	9.74	1	314
Number of Firms	4808	1.19	2.17	0	65.33
Multiple Audiences	4801	1.10	1.14	0	5.67
Multiple Programming Languages	4799	0.58	0.84	0	6.33
Multiple Topics	4802	1.17	1.17	0	6

Regression. We estimate a firm's decision to contribute to an open source project as well as to provide a project leader. We model the decisions as follows:

$$Pr(y_j = 1|\,\mathbf{X}) = \Psi(\mathbf{X}_j\beta + \epsilon_j) \tag{1}$$

Ψ is the cumulative distribution function that specifies the distribution of the error term and $\mathbf{X}_j$ is the vector of the explanatory variables. The subscript j distinguishes between corporate contribution and corporate leadership. We use the same explanatory variables for both dependent variables. They include the license family under which the project is maintained, with Academic licenses as the reference, the number of times an application has been downloaded, the days a project has been posted on SF, the number of multiple topics, audiences and programming languages, the audience for which the software is intended, where we compare the effects relative to the end-user audience. Lastly, the type of operating system on which the software runs enters the regression as well.

Assuming a logistic cumulative distribution function of the error term, equation 1 is:

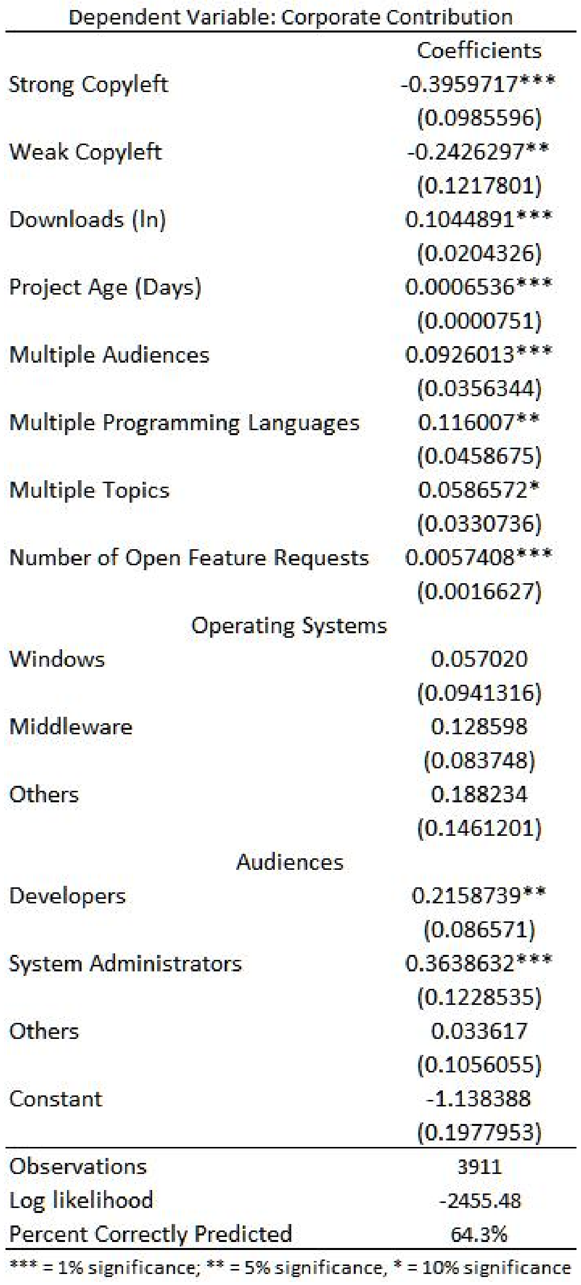
$$Pr(y = 1|\,\mathbf{X}) = \frac{e^{\mathbf{X}\beta+\epsilon}}{1 + e^{\mathbf{X}\beta+\epsilon}}, \tag{2}$$

Table 2 a) presents the estimates for corporate contribution. Companies are more likely to contribute to large projects, i.e. projects that are old and downloaded often. We cannot say anything about the propensity to contribute to, let alone start, new projects - the newest project in our sample is one year old. Still, this lends support to our third prediction. The number of multiple audiences, programming languages and feature requests have also a positive effect on the likelihood of corporate contribution. These findings suggests that firms attempt to control complex projects through contribution, which maintains our second prediction. Moreover, table 2 a) shows that firms are more likely to contribute to projects for technical audiences, such as developers and system administrators. This finding supports our first prediction that firms focus their outsourcing on process rather than product innovations. Furthermore, projects with Academic licenses are more likely to get corporate contributions. This supports our fourth prediction. Unlike Copyleft licenses, Academic licenses allow firms to reconvert applications into proprietary software.

Table 2 b) shows the estimates for corporate leadership. The results are similar to the ones for corporate contribution. We find that one proxy for project size, project age, as well as two proxies for the complexity of the transaction, number of audiences and number of programming languages, are significantly positive, corroborating our earlier findings. We also see that firms are more likely to lead a project for technical audiences rather than end-users. This again lends support to our third prediction. Finally, the results show that firms again favor projects with an Academic license.

Table 2: Regression Results

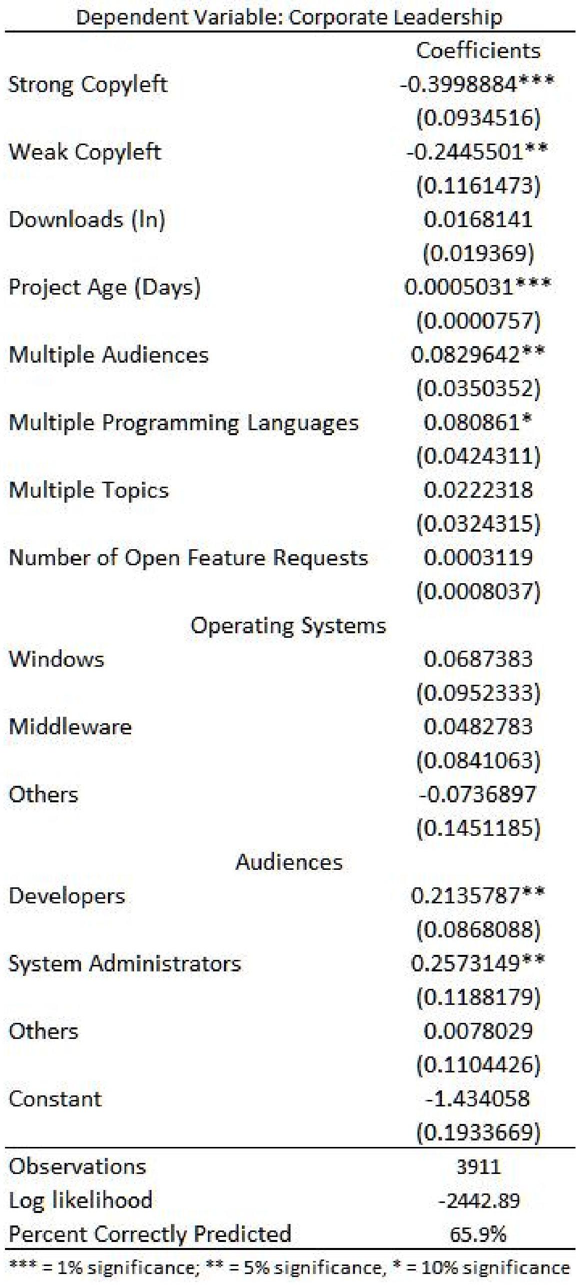
(a) Corporate Contribution



Dependent Variable: Corporate Contribution	
	Coefficients
Strong Copyleft	-0.3959717***
	(0.0985596)
Weak Copyleft	-0.2426297**
	(0.1217801)
Downloads (ln)	0.1044891***
	(0.0204326)
Project Age (Days)	0.0006536***
	(0.0000751)
Multiple Audiences	0.0926013***
	(0.0356344)
Multiple Programming Languages	0.116007**
	(0.0458675)
Multiple Topics	0.0586572*
	(0.0330736)
Number of Open Feature Requests	0.0057408***
	(0.0016627)
Operating Systems	
Windows	0.057020
	(0.0941316)
Middleware	0.128598
	(0.083748)
Others	0.188234
	(0.1461201)
Audiences	
Developers	0.2158739**
	(0.086571)
System Administrators	0.3638632***
	(0.1228535)
Others	0.033617
	(0.1056055)
Constant	-1.138388
	(0.1977953)
Observations	3911
Log likelihood	-2455.48
Percent Correctly Predicted	64.3%

*** = 1% significance; ** = 5% significance, * = 10% significance

(b) Corporate Leadership



Dependent Variable: Corporate Leadership	
	Coefficients
Strong Copyleft	-0.3998884***
	(0.0934516)
Weak Copyleft	-0.2445501**
	(0.1161473)
Downloads (ln)	0.0168141
	(0.019369)
Project Age (Days)	0.0005031***
	(0.0000757)
Multiple Audiences	0.0829642**
	(0.0350352)
Multiple Programming Languages	0.080861*
	(0.0424311)
Multiple Topics	0.0222318
	(0.0324315)
Number of Open Feature Requests	0.0003119
	(0.0008037)
Operating Systems	
Windows	0.0687383
	(0.0952333)
Middleware	0.0482783
	(0.0841063)
Others	-0.0736897
	(0.1451185)
Audiences	
Developers	0.2135787**
	(0.0868088)
System Administrators	0.2573149**
	(0.1188179)
Others	0.0078029
	(0.1104426)
Constant	-1.434058
	(0.1933669)
Observations	3911
Log likelihood	-2442.89
Percent Correctly Predicted	65.9%

*** = 1% significance; ** = 5% significance, * = 10% significance

Conclusion

The question on how to control open source projects is of paramount importance for corporate strategy in OSS. We attempt to shed light on some aspects of this issue. To do so, we relate data on open source projects on SF and developers' corporate background. We then estimate the decision of corporate contribution and corporate project leadership.

With regards to contribution and project leadership, we find evidence that firms opt for projects targeted at audiences related to process rather than product fields. Moreover, firms favor projects with an Academic license. The results suggest as well that firms, because they are more likely to contribute to highly downloaded and mature applications, focus on large projects. More complex projects have a higher likelihood of attracting corporate contribution, which suggests that they attempt to enforce control to reduce the risk of uncertainty.

There are ample points of entry for future research on corporate contribution, control and OSS. First, the analysis of corporate control on OSS could be enhanced with adding firm characteristics and refining the measures for project control. Second, the strategic behavior of firms within open source projects merits further consideration. Do firms cooperate in common projects or do they drive other companies out of projects? Third, the interaction between private and corporate developers raises interesting questions on the development of individual incentives to contribute.

References

Bonaccorsi, A. & Rossi, C. (2003). Contributing to the common pool resources in open source software. a comparison between individuals and firms.

Dahlander, L. (2005). Appropriation and appropriability in open source software. *International Journal of Innovation Management*, 9(3), 259–285.

Dahlander, L. & Magnusson, M. (2005). Relationships between open source software companies and communities: Observations from nordic firms. *Research Policy*, 34(4), 481–493.

David, P. A. & Rullani, F. (2006). Micro-dynamics of free and open source software development: Lurking, laboring and launching new projects on sourceforge. *Stanford Institute for Economic Policy Research Mimeo*.

Gooroochurn, N. & Hanley, A. (2007). A tale of two literatures: Transaction costs and property rights in innovation outsourcing. *Research Policy*, 36, 1483–1495.

Grossman, S. & Hart, O. (1986). The costs and benefits of ownership: A theory of vertical and lateral integration. *The Journal of Political Economy*, 94(4), 691–719.

Henkel, J. (2006). Selective revealing in open innovation processes: The case of embedded linux. *Research Policy*, 35(7), 953–969.

Howison, J. & Crowston, K. (2004). The perils and pitfalls of mining sourceforge. *Proceedings of the International Workshop on Mining Software Repositories (MSR 2004)*, (pp. 7–11).

Kamien, M. & Zang, I. (2000). Meet me halfway: Research joint ventures and absorptive capacity. *International Journal of Industrial Organization*, 18(7), 995–1012.

Klein, P. G. (1998). New institutional economics. *SSRN eLibrary*.

Lakhani, K. & Wolf, R. (2005). Why hackers do what they do: Understanding motivation and effort in free/open source software projects. In J. Feller, B. Fitzgerald, S. A. Hissam, & K. R. Lakhani (Eds.), *Perspectives on Free and Open Source Software* chapter 1, (pp. 3–22). MIT Press, Cambridge.

Lerner, J. & Tirole, J. (2005). The economics of technology sharing: Open source and beyond. *Journal of Economic Perspectives*, 19(2), 99–120. Retrieved from http://ideas.repec.org/a/aea/jecper/v19y2005i2p99-120.html.

Love, J. & Roper, S. (2002). Internal versus external r&d: A study of r&d choice with sample selection. *International Journal of the Economics of Business*, 9(2), 239–255.

Raymond, E. (2001). *The cathedral and the bazaar. Musings on Linux and Open Source by an accidental revolutionary*. O'Reilly.

UNU-MERIT (2006). *Study on the: Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU*. Technical report, UNU-MERIT, the Netherlands.

Veugelers, R. & Cassiman, B. (1999). Make and buy in innovation strategies: Evidence from belgian manufacturing firms. *Research Policy*, 28(1), 63–80.

Wichmann, T. (2002). *Free/Libre and Open Source Software: Survey and Study (FLOSS), Final Report, Part II: Firms' Open Source Activities-Motivations and Policy Implications*. Technical report.

Williamson, O. (1967). Hierarchical control and optimum firm size. *The Journal of Political Economy*, 75(2), 123–138.