Champions of Revealing –
The Role of Open Source Developers in Commercial Firms

This version: January 2008
(earlier version: November 2007)

Joachim Henkel*

Schöller Chair in Technology and Innovation Management
Technische Universität München, Arcisstr. 21, D – 80333 Munich, Germany
phone: +49-89-28925741, fax: +49-89-28925742, email: henkel@wi.tum.de

CEPR, London

Abstract

The link between firms engaging in open source software (OSS) development and the OSS community is established by individual developers. This linkage might entail a principal-agent issue due to the developer's double allegiance to firm and OSS community, and expose the firm to the risk of losing intellectual property. Using both interviews and a large-scale survey, I substantiate the importance of the developer's role. However, neither interview data nor regression analysis show indications of commercially harmful revealing behavior induced by "Free Software ideology." Management, on the other hand, sometimes seems to be overly concerned about openness. I conclude that a more positive stance towards openness will allow firms to better share in the benefits of open innovation processes.

Keywords: Open source software, embedded Linux, employees, motivation, communities

---

* I am grateful to Oliver Alexy, Eric Brousseau, Linus Dahlander, Marc Gruber, Dietmar Harhoff, Jesper Holck, Simone Käs, Inna Lyubareva, Reinhilde Veugelers, Eric von Hippel, Li Yan, and seminar participants at Paris X Nanterre, BETA (Strasbourg), the 3rd International Workshop on User Innovation (MIT), and the 5th EPIP Conference for helpful comments. I thank Mark Tins for valuable help in data collection. All errors are mine.

1 Introduction

The last decade has seen a radical change in the way software is developed and deployed. The Linux operating system and many other types of open source software (OSS) are by now widely adopted by established corporations, both within and outside the IT industry. Nokia and Philips are prominent examples (Engelfriet, 2006; Jaaksi, 2006). They, as numerous other firms, not only adopted existing OSS programs—they also adopted, certainly in some areas, the OSS approach to software development. Contrary to the traditional emphasis put on the protection of intellectual property, this approach implies making developments publicly available, without contractual guarantees of obtaining anything in return. By going far beyond contract-based collaboration (e.g., Arora *et al*., 2001), it can be considered an extreme instance of "open innovation" (Chesbrough, 2003). These fundamental changes pose a challenge to corporations.

In this paper, I focus on a central aspect of this novel open innovation process, namely, the role of the employed OSS programmer. As an employee of his firm and a member of the OSS community, he likely feels loyalty to both entities. This double allegiance can be helpful and even necessary (Dahlander and Wallin, 2006). However, it may also give rise to principal-agent problems, since both information asymmetry and potentially diverging interests are present: A developer will often know better than his supervisor about the benefits and risks of making public a piece of code he has been working on, and he might even reveal it without his supervisor's knowledge. As to the developer's motivation, OSS supporters sometimes subscribe to ideological views on the "freedom of software" (Stallman, 1999; Hertel *et al*., 2003). Quite generally, the personal goals of engineers often differ from those of their employer (Allen, 1988), and an employee's affiliation to a community of practice (Brown and Duguid, 1991; Lee and Cole, 2003) may conflict with the loyalty to her company (Ashforth and Johnson, 2001).

This potential principal-agent problem is compounded by the fact that employed OSS developers hold a key position in the interaction between firm and community. Most firms engaged in OSS development do not disclose all of their developments. Rather, they follow a hybrid strategy, protecting some of their developments while disclosing others as OSS (e.g., Bonaccorsi *et al*., 2006). Even developments based on existing OSS are typically *selectively* revealed (Henkel, 2006). This way, the firm can reap the benefits of open innovation where

these prevail, and can avoid a loss of competitive edge otherwise—provided, of course, the decision to reveal is well advised. This decision is (or should be) guided by general company guidelines, and will in any concrete case be influenced by the individual OSS programmer involved. Being familiar with the technical merits of the code he will often be best positioned to judge the prospective benefits and potential downsides of releasing it to the OSS community.

Participating in open and distributed development of OSS thus requires leaving some discretion to programmers. However, in light of the agency issues sketched above it needs to be analyzed if they abuse this discretion. As participants in public OSS projects they might reveal code which, in the firm's commercial interest, should rather be protected. To avoid such leakage, the employer might forbid revealing of code altogether which, however, would also mean to forgo the potential benefits of the OSS development process (Feller and Fitzgerald, 2002). In addition, what management deems an excessively "open" attitude among its OSS developers may in fact be beneficial for the firm, while management overestimates the risks of openness.

In this paper, I analyze the role of employed OSS programmers in the context described above, focusing on a field of OSS—"embedded Linux"[1]—which is dominated by commercial firms. To obtain a robust picture of the subject of study and to increase validity and reliability of findings, I employ methodological triangulation. Based on 30 in-depth interviews and a large-scale survey (268 respondents) I show that developers have a strong say in the decision to reveal code to public OSS projects, and that excessive or untimely disclosure would be harmful to firms. Interviews as well as survey data on respondents' attitudes point indeed to a potential principal-agent problem between the developer and his firm. However, comparing commercial and non-commercial programmers (those working for non-profit institutions or as hobbyists) shows that the former identify significantly less with the OSS community, and are significantly less ideological about the "freedom of software." More importantly, an analysis of the actual determinants of revealing, using ordered probit regressions, does not support the surmise of a principal-agent problem. Neither the respondent's level of "Free Software ideol-

[1] "Embedded Linux" denotes variants of the OSS operating system which are "embedded" in products ranging from consumer DVD players to control systems for industrial production machines. See Section 4.1.

ogy" nor personal pride in his code are found to be drivers of revealing. Also the level of importance he attaches to reasons to reveal related to the OSS community and his personal benefits have no apparent effect on his contributing behavior. What does have a strong effect on a respondent's revealing behavior, however, is his level of identification with the OSS community. As interview quotes as well as the above findings suggest, this identification is not so much ideological, but most likely due to positive experiences with technical support—and hence beneficial for the firm.

Based on these findings, evidence from interviews, and technical arguments I conclude that the principal-agent problem between the employed OSS programmer and his firm should not be severe. Rather, my data suggest that the programmer often acts as a "champion[2] of revealing," advocating his firm's participation in open and collaborative innovation processes.

The remainder of this paper is organized as follows. Section 2 provides background on commercial OSS development, the motives of individual OSS programmers, and their role as members of two entities. In Section 3, research questions and hypotheses are derived. Section 4 describes the empirical approach. In Section 5, results on the role of the developer, respondents' attitudes, and determinants of revealing are presented. Section 6 concludes.

2 Background

2.1 Commercial OSS development

A number of authors have addressed the question why commercial firms would actively take part in public OSS development.[3] The main potential benefits for firms that have been identified are external development support, setting a standard and enabling compatibility, increasing demand for complements, and signaling technical excellence or good OSS citizenship. Against these benefits, a number of potential downsides must be weighed. Software that is

---

[2] On the concept of "champions" see, e.g., Howell and Higgins (1990).

[3] See Behlendorf (1999), Hecker (1999), Raymond (1999), Feller and Fitzgerald (2002), Lerner and Tirole (2002), Wichmann (2002), West (2003), Bonaccorsi and Rossi (2006), Dahlander (2004), Gruber and Henkel (2005), Gassmann (2006), Henkel (2006), West and Gallagher (2006), and Dahlander (2007). On the issue of collaborative innovation processes, see in particular von Hippel and von Krogh (2003) and Osterloh and Rota (2007).

freely available as OSS can no longer be sold—customers will at best be willing to pay for convenient packaging. Furthermore, such software can be used by competitors, which may imply a loss of competitive advantage. In addition, the original author of the software might lose control over its future development.

Despite these downsides, many firms do engage in OSS development, and likely do benefit from it. In fact, most of the benefits mentioned above are not at odds with wide diffusion of the software, but rather rely on it. In addition, despite the software being OSS firms have various means of protection at their disposal which range from a suitable choice of license (Behlendorf, 1999; Hecker, 1999; Raymond, 1999) to standard means of protection such as copyright, secrecy, lead time, and complementary assets (Teece, 1986; Feller and Fitzgerald, 2002; Dahlander, 2004). The latter are available even in the case of OSS under the GPL, and are consciously and routinely used by commercial firms (Henkel, 2006). Also patents and trademarks may serve as complementary assets (Fosfuri *et al*., 2008). As a result, most commercial players in the field of OSS adopt a hybrid strategy between the extremes of purely open and purely proprietary development (West, 2003; Bonaccorsi *et al*., 2006). They selectively reveal some of their developments in order to benefit from the OSS community (Dahlander and Magnusson, 2005) while protecting others (Henkel, 2006). In this context, it is important to note that "the OSS community" is a rather heterogeneous aggregation of individuals. At the very least, these must be distinguished, for any given public OSS project, into those being paid by a firm for their work on the project ("commercial developers") and those who are not ("non-commercial developers"). As I will elaborate upon in the following subsection and in Section 3, motivations and external conditions should differ strongly between these groups.

2.2 Motivation of OSS programmers

The fact that individuals contribute to OSS without being directly remunerated is one of the most startling aspects of OSS. Various authors have attempted to explain this behavior seemingly at odds with the concept of *homo oeconomicus*.[4] Large-scale surveys (Ghosh *et al*.,

[4] Torvalds considers the fun and challenge of programming as key drivers (Torvalds and Diamond, 2001), while Raymond (1999) stresses the quest for peer recognition. Other authors emphasize economic motivations, in particular adapting software to one's use requirements (Franke and von Hippel, 2003; von Hippel,

2002; Hars and Ou, 2002; Hertel *et al*., 2003; Lakhani and Wolf, 2005) found intrinsic motives (intellectual stimulation, improving skills) ranking highest. They were followed by, in varying order, adapting the software to one's use requirements and motives related to community, reciprocity, and ideology (in particular the view that "information should be free").

Particularly relevant for this paper are two aspects. First, in all surveys the share of respondents who are paid for their development work is considerable. Lakhani and Wolf (2005), for example, identify 40% as "paid," accounting for more than half of total hours spent on OSS development in the sample. Second, there are indications in their survey that some of the motives and actions of paid OSS developers are aligned with their employers' interest, while others are not. The motive ranked as most important is indeed in their firms' commercial interest: "*My contribution creates specific functionality in the code needed for my work*" (55.7%).[5]

This is good news for employers. Still, motives that are independent of or at odds with their interest are non-negligible. Consider the motives ranked second and third in importance, "*The code for this project is intellectually stimulating to write*" (43%) and "*My activity on this project improves my programming skill*" (33.2%). While these motives are not at odds with an employer's interest, they might divert the programmer's effort into interesting, but unproductive activities. Even worse, 30.6% of paid contributors (rank four) ticked "*I believe source code should be open*" as one of their three most important motives, an attitude which should clearly not guide a firm's revealing decisions. Finally, 26.9% (rank five) ticked "*I feel a personal obligation to contribute since I use Free/Open Source software.*" To the extent that such reciprocating improves the firm's image in the OSS community, it may actually be commercially beneficial. Still, *personal* obligations of its employees should not guide a firm's actions. In addition, Lakhani and Wolf find that for 30.7% of the OSS developers who spend work time on an OSS project, the supervisor is *not aware* of their OSS-related activity.

Hence, principal-agent problems are possible and might affect the role of the programmer as the link between company and community. By focusing on a segment of OSS in which

---

2001; Osterloh and Rota, 2007), and signaling competence and skills to the job market (Lerner and Tirole, 2002; Franck and Jungwirth, 2003; Lee *et al.*, 2003).

[5] Percentages refer to the share of paid contributors in the sample who picked the respective motive as one of the three most important motives out of eleven to contribute to OSS projects.

most contributions are commercial, my data allows to analyze this role in detail. In particular, I can relate a respondent's revealing behavior to his attitudes and stated reasons to reveal.

2.3 Organizational identification – Between company and community

If a firm benefits from its OSS engagement depends to a large extent on its employed OSS developers. While firm policies certainly matter, it is, as my data show, often the individual programmer who advocates and possibly also decides about revealing a particular piece of code to the OSS community or not.[6] Cases of such "champions of revealing" are reported by Henkel (2004a) who describes the cases of "CEPS" ("Cisco Enterprise Printing System", earlier mentioned by Raymond, 1999) and "OpenAdaptor", a middleware developed by the investment bank Dresdner Kleinwort Wasserstein. These software packages were released as OSS in 1997 and 2001, respectively. In both cases, the idea of going OSS was conceived and championed by the maintainers of the software.

The programmer also determines his firm's reputation within the respective OSS project by submitting more or less helpful and competent comments and code. Both revealing the right code and having a good reputation are preconditions for benefiting from external development support, and also influence if diffusion-related benefits are realized. In turn, revealing the wrong piece of code can imply a loss of competitive advantage, may reveal an infringement on some third party intellectual property right, or may restrict the firm's options for commercializing its code more than is acceptable. In addition, inappropriate behavior by the developer in the OSS community will do damage to the firm's reputation.

To a good extent, monitoring—and even the mere possibility of it—helps to mitigate these risks. Contributions to mailing lists and code repositories of public OSS projects are preserved for a long time, and can be checked when the suspicion of incorrect behavior arises. In addition, a firm can keep logs of all data leaving its network. These rather broad monitoring possibilities distinguish the present case of disclosures from, e.g., information sharing in strategic alliances (Hamel *et al*., 1989).

---

[6] Similar discretion on lower hierarchical levels with respect to information sharing has been reported by Hamel *et al*. (1989) for the case of strategic alliances, and by Schrader (1991) and Dahl and Pedersen (2004) for the case of information trading (see also von Hippel, 1987). As Hamel *et al*. (1989, p. 136) write, "Top management […] sets the legal parameters for exchange. But what actually gets traded is determined by day-to-day interactions of engineers, marketers, and product developers."

Still, two limitations to monitoring, and two sources of information asymmetry, persist. First, within the leeway defined by general firm guidelines the employed OSS programmer himself will often be the best person to judge if revealing a certain piece of code is harmful or beneficial for his firm, since he likely is best informed about the code's technical merits, its uniqueness, and its potential use by other firms. Second, he could still submit code anonymously from his private PC. Such behavior was reported in the case of OpenAdaptor mentioned above (Henkel, 2004a). Programmers working for competitors of Dresdner Kleinwort Wasserstein were typically barred by their labor contracts from sharing any of their code with anyone. As a result, the maintainers of OpenAdaptor received "a lot of anonymous contributions." Interestingly, these illegitimate disclosures were most likely *beneficial* for the respective firms, since they led to future versions of the software being improved or even adapted to specific needs of the submitters. In contrast, a competitive disadvantage from improving a competitor's middleware appears unlikely. It thus seems that the legal departments had taken an overly restrictive stance on openness.

The above implies that commercial OSS programmers' motivation, attitude, and loyalties are of central importance in our context. In particular, the issue of double allegiance, or multiple identities, arises. As argued above, these multiple identities of employed OSS programmers are indispensable if the firm is to take full advantage of OSS, and firms may even employ individuals to act as dedicated boundary spanners between them and an OSS community (Dahlander and Wallin, 2006). However, in a particular situation these identities may be conflicting. Following Ashforth and Johnson (2001), a commercial OSS programmer's identities as an employee on the one hand and as a member of a public OSS project on the other hand are instances of "cross-cutting" identities. Their relative *salience* (e.g., Stryker, 1980) determines if the programmer, in a situation of conflict, acts in his firm's or his community's interest. An identity's salience, in turn, is determined by its *situational relevance* and *subjective importance* (Ashforth, 2001). Situational relevance of the developer's "OSS identity" will likely be highest (and thus most relevant) when he needs to decide about sharing some code or keeping it secret, since sharing is at the heart of the OSS idea. Its subjective importance, in contrast, is relatively stable over time, and can be captured by survey questions.

3 Research questions and hypotheses

The role of employed programmers at the interface between their firm and the OSS community is determined by an external and an internal factor. The external factor are *firm policies* with respect to revealing, in particular the discretion the developer has. The internal factor are his personal *attitudes* towards revealing and OSS. Both are expected to influence his *revealing behavior*. In the following, I formulate research questions and hypotheses regarding the above three issues.

Firm policies. How strongly a developer's attitudes and behavior impact his firm in the context of commercial OSS development depends on the discretion that his firm leaves him, and on the firm's general policy towards revealing. Does he have strict guidelines to follow, does he discuss each case with his supervisor, or is the decision largely left to his own discretion? Is an official firm policy on revealing in place, and is it supportive or restrictive? And, when revealing does occur, is the supervisor aware of it?

Attitudes. An employed OSS programmer belongs both to his company and to the OSS community. As laid out in Section 2.3, the relative salience (Stryker, 1980) of these two identities determines in whose interest he acts in cases of conflict, that of his company or that of the OSS community (Ashforth and Johnson, 2001). These interests will often be aligned, but will differ in some cases. The risk that the programmer, in such cases, acts in a way harmful to his company is arguably higher the higher the subjective importance of his "OSS identity" is to him (Ashforth, 2001). This importance is derived from his identification with the OSS community and the extent to which he shares positive ideological views, in particular on the "freedom of software." While it is hard to say which absolute level of OSS identification and ideology is critical for an employer, the attitudes of *non-commercial* programmers working in the same field provide a natural benchmark. There are good reasons to assume that the two groups differ. For commercial programmers, their job provides a motivation to work on OSS, which is absent for non-commercial developers. Hence, in order to have a sufficiently strong *overall* motivation for working on OSS, the threshold level of *non-job-related* motivation needs to be higher for non-commercial than for commercial programmers. Under plausible assumptions[7] it follows that also the *average* level of non-job-related motivation should be

[7] This statement holds true when the shape of the distribution of non-job-related motivation levels does not differ too much between the two groups of programmers.

higher for non-commercial programmers. The most relevant sources of such motivation in the context of principal-agent issues are, as discussed in Sections 2.2 and 2.3, their attitude towards "Free Software ideology" and their identification with the OSS community.[8] In addition, commercial OSS programmers will often also work on proprietary software since it constitutes a part of the business model of many firms developing OSS (e.g., West, 2003; Dahlander, 2004; Fosfuri *et al*., 2008). In order to reduce cognitive dissonance (Festinger, 1957), they should thus have a more positive attitude towards proprietary software and less ideological views on OSS. I thus posit the following two hypotheses.

> *H1a: Commercial programmers identify less strongly with the OSS community than non-commercial programmers.*
>
> *H1b: Commercial programmers share positive ideological views on the "freedom of software" to a lesser extent than non-commercial programmers.*

Reasons to reveal. Just as their non-commercial counterparts, commercial OSS programmers reveal code partly out of motives not related to their firm's benefits, as reviewed in Section 2.2. These can be divided into community-related motives and those linked to the programmer's personal benefits—or, as Simon *et al*. (1998) phrase it in the context of social movement participation more generally, motives based on collective identification and cost-benefit calculations. Both types of reasons should matter more for non-commercial than for commercial programmers. As to community-related motives, the argument proceeds along the same lines as for attitudes. Also with regard to personal reasons to reveal code (learning, signaling to the job market), a similar logic should apply: given that the motive "code needed for my work"—by far the most important one for commercial programmers in the study by Lakhani and Wolf (2005)—is irrelevant for non-commercial programmers, other motives need to

---

[8] On the motives of (commercial and non-commercial) OSS programmers see Ghosh *et al*. (2002), Hars and Ou (2002), Hertel *et al*. (2003), and Lakhani and Wolf (2005). On motivation based on collective identification and how it interacts with other sources of motivation, see Klandermans (1997) and Simon *et al*. (1998).

be of higher relevance for them. For "learning" in particular, this consideration is clearly supported by Lakhani and Wolf (2005).[9] These considerations lead to the following hypotheses.

> *H2a: For commercial programmers, community-related reasons to reveal matter less than for non-commercial programmers.*
>
> *H2b: For commercial programmers, reasons to reveal related to personal benefits (learning, signaling) matter less than for non-commercial programmers.*

Revealing behavior. I now turn to commercial programmers' actual revealing behavior. After all, even an employee's most radical ideological views on the "freedom of software" matter little for a firm as long as they do not impact his actions. Using qualitative data from interviews I address the question if and how programmers submit code without their employer's approval or even in breach of explicit firm policies. Next, I analyze what drives revealing. Here, it is of central importance if more code was submitted by programmers (a) who share a more ideological attitude towards "Free Software" and (b) who take more pride in their code. In these cases, (a) ideology and (b) personal vanity would be drivers of revealing—and the firm would have some reason to be concerned about its developers. The effects of these and several other variables are important for understanding the relative subjective importance of the programmer's "cross-cutting identities" (Ashforth and Johnson, 2001). Note that the *direction* of the effects dealt with in Hypotheses H3 to H8, developed below, is mostly not surprising, which is also why deriving the hypotheses is not too sophisticated. However, it is not at all clear if they are *significant* since, e.g., developers might simply ignore and circumvent restrictive firm policies.

As to firm policies, it is very plausible that a restrictive firm policy towards revealing code should impact a developer's contributions negatively, while an encouraging policy should have a positive effect (Hypothesis H3, see below). A person's identification with the OSS community has been found to influence the extent of that person's contributions to public OSS projects positively (Hertel *et al*., 2003) as, more generally, participation in collective action is furthered by identification (Simon *et al*., 1998). This leads to Hypothesis H4. A fur-

---

[9] For the motive of "signaling," also a countervailing argument could be made: contributing to OSS as a commercial programmer might provide a more positive signal to the job market than contributing as a (possibly unemployed, in any case apparently under-worked) hobbyist.

ther potential motivator for contributing is "ideology" related to the "freedom of software." A political position supportive of the "freedom of software" leads, nearly by definition, to the hypothesis that a developer who shares stronger ideological views on Free Software contributes more (H5).[10] Raymond (1999) argued that OSS developers might share their work because they strive for reputation among their peers. To that purpose, code to be proud of should be best suited, and programmers who take pride in their work should be most likely to use it to impress others. This implies Hypothesis H6. Finally, one should expect a developer to contribute more the more importance he attaches to reasons to reveal related to the OSS community (H7) and his personal benefits (H8). In sum, the following hypotheses have been derived:

A developer contributes more frequently code to public embedded Linux projects …

H3: … the more supportive firm policies are towards revealing.

H4: … the stronger his identification with the OSS community is.

H5: … the more he shares positive ideological views on Free Software.

H6: … the more pride he feels for his work on embedded Linux.

H7: … the more important community-related reasons to reveal are for him.

H8: … the more important personal benefits are for him as reasons to reveal.

4 Empirical approach

4.1 Embedded Linux

I study the behavior of OSS programmers in commercial firms for the case of "embedded Linux." The term denotes versions of the open source operating system Linux that are "embedded" in products which are, unlike PCs or laptop computers, dedicated to a single purpose. Examples range from consumer DVD players over cell phones and payment terminals to control systems for industrial production machines. Using Linux in such devices has become very common over the last years (Webb, 2002; 2006). It is today one of the three most widely used embedded operating systems (VDC 2004).

[10] Note that this hypothesis is not at odds with H1b. While H1b hypothesizes about the difference of ideology levels *between* commercial and non-commercial programmers, H5 concerns the effect of different ideology levels *within* the group of commercial programmers.

Since devices and thus the requirements to the operating system are extremely heterogeneous, there is no standard version of embedded Linux. Rather, "embedded Linux" refers to various versions of Linux that are, in one way or another, adapted to embedded systems. Examples for such adaptations are the RTAI real-time module ("Real-Time Application Interface"), the toolkit busybox, the shrinked C library uclibc, and processor-specific code.

Heterogeneity of embedded Linux marks an important difference to more homogeneous OSS packages such as standard Linux. It implies, among other things, that free riding on developments performed by others is restricted and that standardization is much less relevant. Firms in this field thus face a rather different situation than, e.g., the Linux distributor Red Hat.

New developments based on existing embedded Linux code are to be regarded as "derived work" in the sense of the GPL license, which governs the use of Linux. This implies that, by the time a device containing embedded Linux comes onto the market, the source code of the version of Linux it contains must be made available to all buyers. Furthermore, patents play hardly any role in this field.[11] Still, considerable leeway does exist with respect to revealing or protecting one's developments for embedded Linux (Henkel, 2006): It can be given only to paying customers, who often prefer to keep the code secret; revealing can be delayed until the device comes to market, which is on average 18 months after the code was written; and by using "binary loadable modules" for drivers, revealing can be avoided altogether.

Embedded Linux is well suited for the question at hand since, as will become clear below, most contributions to public embedded Linux projects come from commercial firms. This implies that the "OSS community" referred to in the present context mainly consists of commercial programmers.

4.2 Interviews

A total of 30 in-depth interviews were conducted by the author in the period between May 2002 and June 2003. Of the interviewees, the largest group (13) worked for software firms

---

[11] Fosfuri *et al*. (2008) analyze the use of patents, among others, as complementary assets by firms releasing software products under OSS licenses. Yet, in the specific field of embedded Linux patents do play only a minor role. This is due to the fact that, for hardware firms, their hardware serves as a protecting complementary asset, while most software firms perform commissioned development instead of selling standardized software products. Others (e.g., Montavista) rely on brand and warranties as complementary assets.

specializing on embedded Linux; six were employed by hardware manufacturers using embedded Linux; seven were industry experts; and four were indirectly dealing with embedded Linux, mostly as sellers of competing products. Most interviews (28) were conducted as oral conversations, two by e-mail. Of the 28 oral interviews, 18 were electronically recorded; in 10 cases handwritten notes were taken. The average length of the interviews was 53 minutes. They were transcribed and analyzed using the software "AnSWR" to perform a qualitative content analysis (Strauss and Corbin, 1990). During the process of open coding, a hierarchical category scheme was developed consisting of 137 categories. In total, 657 relevant text segments were identified and assigned to one or more categories.

The focus of the interviews was on the peculiarities of the open source development process in an environment dominated by commercial firms, not hobbyists. In particular, it addressed collaboration between OSS programmers employed by competing firms.

4.3 Survey

Set-up and respondents

A survey complementing the interviews was conducted between November 2003 and March 2004, using a web-based questionnaire. The survey, targeting developers, was pre-tested and then advertised on web portals and mailing lists dedicated to embedded Linux development. It contained 115 questions and yielded 268 valid responses.

The clear majority of participants (73.5%) indicated that they develop embedded Linux as part of their job working for a commercial firm. Of these, two thirds work for hardware firms (51.1% in total; 42.5% for device manufacturers, 8.6% for component manufacturers), about one third (22.4%) for software firms. In the following, I refer to these respondents as "commercial" developers.

The remaining 26.5% of respondents identified themselves as "hobbyist" (15.3%) or as working for "University or other non-profit research organization" (11.2%). Hobbyists in the sample work less hours per week than commercial programmers (contributing only about 7% of total hours per week in the sample), and will likely be overrepresented in the survey due to self-selection. Hence, hobbyists indeed play only a minor role in embedded Linux development. Still, even though relatively small, the group of non-commercial programmers provides a very useful benchmark for analyzing characteristics of commercial developers.

Variables

In Section 5.2, I will first provide a descriptive analysis of firm policies. To identify these policies, participants were offered five statements and were asked to tick those that apply (see Table 1). The dummy variables coding an encouraging and a restrictive policy were subtracted to yield the index "PolSupportiveIndex," to be used in the regression. In addition, participants were asked how often their supervisor was aware of their sharing code, both generally and with direct competitors. Responses are described in Section 5.2.

Insert Table 1 about here

Next, I will analyze differences between commercial and non-commercial programmers with respect to their attitudes. The latter were measured as the level of agreement to various statements, on a scale from -2 ("strongly disagree") to +2 ("strongly agree"), see Table 2. Identification with the OSS community (variable name "OSSidentification", H1a) is measured as the level of agreement to the statement, "*I identify with the open source community*". The extent to which the respondent shares ideological views on the freedom of software ("FreeSWideology", H1b) is operationalized by the level of agreement to "*Free Software matters because all freedom matters.*"[12] Finally, also the level of pride taken in one's work on embedded Linux is asked for ("Pride"). The last part of the descriptive analysis addresses personal and community-related reasons to reveal (see Table 3), the relevance of which was again measured on a 5-point scale.

Insert Table 2 about here

Insert Table 3 about here

In the multivariate regression, I use the frequency of code contributions to public embedded Linux projects as the dependent variable, measured on a five-point scale (*never, once*

[12] "Ideology" is, both generally and in the context of OSS and Free Software, a complex object. I focus on the specific aspect of the "freedom of software," since by definition this element of ideology should impact directly on the respondent's attitude towards revealing (which amounts to "making software free"). Given that the term "Free Software" is a hallmark of the Free Software Foundation (FSF) which is known for its ideological stance (Stallman, 1999), the statement should indeed capture ideological views of the respondent.

a year, once a month, once a week, several times per week).[13] Descriptive statistics are provided in Table 4.

Insert Table 4 about here

Explanatory variables not yet described above are the following (see Table 5). "ReasonsCommunity" and "ReasonsPersonalBenefits" are indices comprising several items describing motives to reveal related to the OSS community and to personal benefits.[14] In addition, nine control variables are used. Control variables related to the developer are the number of years he has been developing embedded Linux (DevelYearsEL), the total number of hours per week he spends on embedded Linux development (ELTimePerWeek), the share of this time that is spare time (ShareSpareTime), and a dummy variable indicating if the respondent's experience in OSS development is longer than that in embedded software development (OSSbeforeEmbed). Control variables describing characteristics of the employing firm are two dummy variables for size (SizeMedium: 11 – 200 employees; SizeLarge: more than 200 employees), the number of years the company has been developing embedded Linux (CompanyYearsEL), and two dummy variables indicating the type of firm (TypeDeviceMfr: device manufacturer; TypeComponentMfr: component manufacturer; reference group: software firm). Correlations between all explanatory variables are provided in Table 6.

---

[13] Ideally, one would like to have a quality-weighted measure of the quantity of code that is submitted. However, questions about the quality of submitted code would have been hard to answer, and the answers even harder to compare between respondents. Also a pure quantity measure such as lines of code is not satisfying since large pieces of code may be submitted that have been recycled from some other software program.

[14] "ReasonsCommunity" comprises the level of agreement to the statements, "*I reveal code because I consider it fair to give back to the community, since the company benefits from it*" and "*… because, in the long run, you only get something when you gave something before*." In "ReasonsPersBenef" the items "*… because I get feedback on my code, which improves my personal skills*", "*… because I get feedback on my code, which improves my performance at my current job*", "*… because I get recognition in the open source developer community for my contributions*", and "*… because I demonstrate my skills to future employers*" enter. Factor analysis, using principal component analysis and varimax rotation, clearly yields these two factors, with Cronbach's α of 0.52 (ReasonsCommunity) and 0.84 (ReasonsPersBenef), respectively. Given the rather similar meaning of the items entering ReasonsCommunity, the alpha value is surprisingly low. However, using these items separately in the regression yields coefficients which are just as insignificant. This shows that no relevant results are affected by using the index versus using the individual items.

Insert Table 5 about here

Insert Table 6 about here

5 Results

In this section, I first provide qualitative evidence concerning the role of commercial OSS programmers. I then present a descriptive analysis of survey data on programmers' roles and attitudes, before analyzing correlates of revealing using regression analysis. Both 5.1 and the beginning of 5.2 are thus dealing with research questions which are not linked to hypotheses. Testing Hypotheses 1 to 8 is the subject of the second part of 5.2.

5.1 Qualitative evidence

In the following, I provide qualitative evidence for four propositions: (1) Firms selectively protect their embedded Linux-related developments. Hence, a "too open" behavior by its programmers would be commercially harmful. (2) Employed OSS programmers share information in public OSS projects partly out of personal motives. (3) Management is not always informed about this sharing and has broad, but nonetheless limited means of monitoring it. (4) Management may overestimate the risk of critical code leaking out.

(1) Quotes (a) to (c) show that firms may selectively protect their developments by delaying disclosure. Going beyond delaying, firms may also try to avoid revealing altogether, as illustrated by quotes (d) to (f).

> *(a) "I observe competition between embedded Linux vendors. For certain periods, they withhold their developments from each other, and after a while nonetheless do share them."* (Interviewee 1, device manufacturer, Europe)[15]

> *(b) "People certainly sometimes ponder, do I put the code on the web right now or is there a chance to make another buck from it?"* (Interviewee 2, embedded Linux vendor, Europe)

[15] When necessary, the quotes have been translated to English by the author.

(c) "Very often it is the case that patches are simply held back for 2 to 3 months." (Interviewee 3, embedded Linux vendor, Europe)

(d) "In the embedded arena we have a tough competition situation and it has become a much more commercial Linux market […] For that reason, people watch a bit more carefully which information they give away which could help their competitors." (Interviewee 3, embedded Linux vendor, Europe)

(e) "Other times they might have value added components that differentiate their product from other people's product, that they don't want to invest the time and money developing that special interface and then give that away to their competitors." (Interviewee 4, embedded Linux vendor, US)

(f) "Imagine [Firm X, a large electronics manufacturer] has developed boards which, e.g., can serve 48 UMTS channels. These are specific boards […], based on some standard processor. When for these boards and their periphery a driver is developed, then this driver contains specific, [Firm X]-internal protocols. When you put these drives as open source on the web, [Competitor Y] can come along and replicate the device easily. In that case, we have a lot of intellectual property on kernel level." (Interviewee 3, embedded Linux vendor, Europe)

(2) Most interviewees contended that embedded Linux developers are rather pragmatic about the "freedom of software." Still, they considered that "Free Software ideology" does play a certain role. Also other personal motives such as pride and altruism matter, as the following quotes (g) and (h) illustrate.

(g)"Clearly there are humans behind it and to a certain extent what counts most in this kind of open development is also, well, the feeling of showing up and making yourself known." (Interviewee 5, device builder and user, Europe)

(h) "It is not so much that the companies themselves are collaborating to help people. It's that individuals who are intelligent and interested in that area and have the knowledge and skills tend to be employed by the companies, and they are willing to help and share." (Interviewee 4, embedded Linux vendor, US)

(3) In quote (i) below, a development group leader at a large electronics corporation explains information sharing by OSS developers in his group. The interviewee, not at all ideological about OSS, concedes that upper management is unaware of this sharing. Later on, he points out that management has only limited means of monitoring information sharing by its employees (quote j).

(i) Question: *"Would developers support each other, even if working for competing firms?"* Answer: *"Yes, that's the case."* Q: *"And management does not object?"* A: *"They don't know about it at all."* Q: *"And if they knew?"* A: *"Then we would have to explain that we really don't compete on this dimension. This is not a differentiating feature. There are benefits to standardization."* (Interviewee 1, device manufacturer, Europe)

(j) "Anyway, you can't control such contributions to mailing lists – the developers would do it at home from their private PCs. My personal opinion: we shouldn't control such contributions." (Interviewee 1, device manufacturer, Europe)

(4) Quotes (i) and (j) may be somewhat disquieting for management. However, the interviewee considers the developers' contributions, even if clandestine, beneficial for the firm (see quote (i)). The problem he perceives is that management might not understand the benefits of informal collaboration. A similar assessment is reflected in the final quote (k):

(k) "It is normal to ask a question about a particular programming problem, and to get an answer, even though the other programmers work for competitors. This information exchange is more common among technical people than among management, maybe because management knows less clearly where the critical points are." (Interviewee 6, technical account manager at embedded Linux vendor, Europe)

Summarizing, the interviews and in particular the quotes given above support the propositions put down at the outset of this section. In the next section, these qualitative insights are complemented by quantitative results.

5.2 Survey

The role of developers

Regarding firm policies (see Table 1), the statement "*There is no official policy*" received by far the highest level of agreement (46.7% of all respondents working for commercial firms, N = 197). While it might be that firms deliberately leave some ambiguity in order to react to commercially harmful acts of revealing in a flexible way, the more likely explanation seems to be that setting up an official policy has, so far, not been considered.

The second highest level of agreement (34.0%) received the statement "*I make suggestions as to what we could make public, and discuss each case with my supervisor*." The de-

veloper in this case seems to act as a "champion of revealing"—trying to convince his supervisor of the benefits of revealing on a case-by-case basis.

For about a quarter of respondents (25.4%), revealing source code is in their own responsibility. Since the individual developer may often be in the best position to judge the pros and cons of revealing a particular piece of code (see quote (k) in Section 5.1), this policy makes sense—provided, of course, the developer acts in the firm's interest. Finally, 22.8% of respondents described their firm's policy towards revealing as "*encouraging*", 16.8% as "*restrictive*". It is interesting to note that 28% of respondents from large firms reported a restrictive firm policy, in contrast to only 12.2% from small and medium-sized firms.

The above findings underline the importance of the developer's role. When it is the developer who decides about revealing or suggests it to his supervisor, the firm relies on the developer's judgment for identifying suitable code. This is firm policy in 49.7% of all cases (98 of 197 commercial developers), and in about half of these cases (25.4%) even the final decision to reveal is left to the developer.[16] Furthermore, it seems plausible that also in many of those cases where there is no official firm policy (and where none of the two statements above had been ticked, 26.4% of respondents) the developer suggests or even decides about revealing.

Supervisors' awareness of code sharing by their developers is relatively high. More than half (50.7%) of all commercial respondents who answered this question (N = 169) stated that their supervisor was "always" aware of it. "Often" or "sometimes" was ticked by 41.4%, and only 6.6% said that their supervisor was "rarely" or "never" aware. In the more critical case of sharing code with *direct competitors*, a much higher share of respondents (71.3% of N = 136) said that their supervisor was "always" aware of their code sharing. 16.9% stated that the supervisor was "often" or "sometimes" aware, and 11.8% ticked "rarely" or "never". While a social desirability bias in these responses can not be excluded, it will likely not be strong since anonymous and confidential treatment of responses was credibly assured.

These findings show that the developer is indeed of central importance in defining his firm's role in public OSS projects. In at least half of the cases in the sample, it is him who suggests or selects code to be made public—a crucial choice both for exploiting the opportu-

---

[16] Of all 197 commercial programmers, 67 (34%) stated that they make suggestions and discuss them with their supervisor, and 50 (25.4%) said that revealing is in their own responsibility. These numbers do not add up to 59.4% since 19 respondents (9.6%) ticked both statements (see Table 1).

nities of public OSS development and for avoiding its risks. The number of respondents who do not inform their supervisors when sharing code with direct competitors is relatively small (11.8%), but non-negligible. This finding may either mean that (a few) developers constitute potential leaks, or that they—consciously, but in the firm's best interest—disregard management's overly restrictive rules on revealing. In order to better understand how the developer uses his discretion a look at his attitudes and personal reasons to reveal is warranted.

Developers' attitude towards OSS

The statement "*I identify with the open source community*" received a relatively high level of agreement from both groups (see Table 2). Still, agreement from commercial developers is significantly lower: "I somewhat agree" or "I strongly agree" was ticked by 81.4% of commercial and 92.3% of non-commercial developers, and a Mann-Whitney test rejects the null hypothesis of equal medians on the 10% level ($p = 0.06$). Hence, hypothesis H1a is supported.

Identification with the OSS community may be entirely pragmatic, based on positive experiences with external development support. In contrast, subscription to the view that "*Free Software matters because all freedom matters*" does reveal an ideological attitude. The respective levels of agreement to this statement are lower than the levels given above, and differ more clearly between commercial developers (64.5% "somewhat agree" or "strongly agree") and non-commercial developers (78.8%). A test on differences of medians is highly significant ($p = 0.003$). Thus, also H1b is supported. Commercial OSS programmers *do*, on average, identify with the OSS community are are even slightly ideological about the "freedom of software," but significantly less so than the reference group.[17]

---

[17] With respect to H1a and H1b, a very interesting question arises: are the differences due to learning or to selection? In other words, do OSS programmers adopt over time, after they became employed to work on embedded software, a less ideological and less community-oriented attitude (see Westenholz, 2003)? Or do those OSS programmers become employed who are less ideological and less community-oriented in the first place (see Alexy and Leitner, 2008)? While a comprehensive analysis of this question is beyond the scope of this paper, some preliminary results are the following. With respect to FreeSWideology, a selection effect seems to dominate since the variable is not correlated with the number of years the respondent has been developing embedded Linux (variable "DevelYearsEL") which, for commercial developers, is a good proxy for being exposed to commercial OSS work. Furthermore, the two groups of commercial developers who had (group C1) resp. had not (group C2) worked on OSS before working on embedded software do not differ significantly w.r.t. FreeSWideology. In contrast, for the variable OSSidentification the groups C1 and that of

As reviewed in Section 2.2, also personal pride taken in one's software developments might matter for a person's revealing behavior. However, since it appears less clear than in the cases above if the two groups should differ in this respect, I did not posit a hypothesis. As Table 2 shows, both groups show pride in their work, and do not differ significantly.

Regarding respondents' community-related and personal reasons to reveal, differences between the two groups of respondents are much less clear. As Table 3 shows, agreement to community-related reasons to reveal ("*I reveal code because I consider it fair to give back to the community*" and "*I reveal code because, in the long run, you only get something when you gave something before*") is only slightly higher for non-commercial developers, and the difference is insignificant. H2a is thus not supported. As to personal motives, agreement to "*I reveal code because I get feedback on my code, which improves my personal skills*" is significantly higher for non-commercial programmers ($p = 0.028$), while for the motive to "*demonstrate my skills to future employers*" no significant difference can be found. H2b thus receives mixed support. Regarding the absolute levels of importance across both groups, I find that the motive "*to give back to the community*" receives the highest agreement (mean / median = 1.62 / 2), followed by "*feedback on my code*" (0.96 / 1) and reciprocity (0.90 / 1). Signaling to potential future employers matters little for both groups (0.32 / 0).

Correlates of revealing behavior

Results of the multivariate analysis of respondents' code contributions are shown in Tables 7 and 8.[18] I employ ordered probit regression, with three specifications. Specification (1) contains the index PolSupportiveIndex, (2) contains the individual variables PolEncouraging and PolRestrictive. Since these can partly be explained by the firm's basic characteristics as described by the control variables (size, type, experience with embedded Linux), alternative

---

non-commercial programmers do not differ, while both differ strongly significantly from the group C2. In addition, for the latter group OSSidentification is highly significantly correlated with DevelYearsEL (rho = 0.27, $p = 0.002$). In sum, these findings do suggest a learning effect – however, not for OSS developers turned (commercial) embedded Linux programmers, but rather for commercial developers of embedded software who later started working on OSS, and learned about the benefits of the OSS community.

[18] Since the categories *once a week* and *several times per week* of the dependent variable contained relatively few responses (7 and 9, respectively, corresponding to 6.0% / 7.7% of all 117 observations in the regression), these categories were merged in an alternative specification. Since the regression results remained nearly unchanged I report only the specification using all five categories.

specification (3) leaves out the policy variables.[19] Variables insignificant in the full specifications (1a), (2a), and (3a) were successively eliminated, leading to the respective final model. A Wald test confirms joint insignificance of the eliminated variables in all three cases.[20]

Insert Table 7 about here

Insert Table 8 about here

As expected, a supportive policy towards revealing has a positive effect on the frequency of code contributions in specifications (1a) and (1b) which is significant on the 1% level, confirming H3. Marginal effects are reported for Table 7.[21] Breaking up the index into its constituents (specifications 2a, 2b) yields a significant effect (5% level) of an encouraging policy. A restrictive policy, while it does have a negative effect of about the same size just fails to be significant at the 10% level.

Of the three variables relating to the developer's attitudes, "FreeSWideology" and "Pride" do not exhibit a significant effect in any specification. Thus, for H5 and H6 the null hypothesis that an ideological attitude resp. the developer's pride in his work have no effect

---

[19] Table 6 shows that having an encouraging company policy towards revealing is significantly correlated with the number of years the firm has been developing embedded Linux (rho = 0.26, p = 0.005). Also, having a restrictive policy is significantly correlated with being a large firm (rho = 0.162, p = 0.082).

[20] The ordered probit model implicitly assumes that the effects of the explanatory variables are identical at each cut-off point between categories (the "parallel regression assumption"). To check if this assumption is justified, I ran generalized ordered probit regressions using the GOPROBIT command in STATA 9. Due to issues with convergence, the parallel regression assumption was relaxed only for the three most important explanatory variables (those related to H4 – H6: OSSidentification, FreeSWideology, Pride). For none of these variables, the hypothesis of identical coefficients for all cut-off points could be rejected at the 10% level. Hence, the use of an ordered probit model instead of a generalized ordered probit model is justified.

[21] The columns "marg. eff." allow a sensitivity analysis of predictions. They are to be interpreted as follows. When, for a hypothetical new observation, all variables have values as indicated in column "x" (these are the sample means), then increasing the value of a particular variable (row) by a small amount Δ (or, for dummy variables, $\Delta = 1$) increases the probability of that observation being in the n-th ordered Probit category by Δ times the value given in column "marg. eff. n." For example, an increase in OSSidentification by Δ implies an increase of the probability of the observation being in category 5 ("code contributions several times per week", see Table 4) by Δ times 1.1%.

on the frequency of his contributions can not be rejected. In contrast, the level of identification with the OSS community is significant in all specifications (1% level or 5% level). The effect size is large: an increase by two points has a stronger effect than a switch from no policy to an encouraging policy. H4 is thus supported.[22]

This difference between the test results concerning "OSSidentification" and "FreeSWideology" is of central importance. Identification with the open source community may be completely non-ideological, arising from positive experiences with development support in the open source process. In contrast, subscription to the view that "*Free software matters because all freedom matters*" does reveal an ideological attitude. Insignificance of "FreeSWideology", and significance of "OSSidentification", thus shows that not ideology, but more likely positive experiences with the open source community drive revealing on the developer's level. This finding is highly relevant for the developer's employer: the firm can be somewhat reassured that its programmers' revealing behavior is driven not by (potentially harmful) ideology but by positive experiences with the open source process.[23] From the fact that "Pride" does not exhibit a significant effect, a similar conclusion can be drawn: just as ideology, personal vanity does not appear to be a driver of revealing.

Finally, also "ReasonsCommunity" and "ReasonsPersBenef" fail to be significant in all specifications, both individually and jointly. The interpretation is analogous to that given above for "FreeSWideology" and "Pride". Since these reasons are not related to the firm's benefits but to those of the community and the programmer himself, respectively, their being significant would be a worrying finding for firms. In turn, their insignificance is comforting for employers: even if a programmer attaches more importance to these reasons, he does not contribute more frequently to the OSS community.

---

[22] One might conjecture that insignificance of "Pride" and "FreeSWideology" is caused by the fact that both are correlated with "OSSidentification". However, their insignificance persists when "OSSidentification" is dropped from the list of explanatory variables.

[23] Such positive experiences are captured in statements from interviewees: "If it's a generic piece of code that is published as free SW, other people in the community will use that piece of code, find bugs, and will continue to improve it. The benefit we see there is lower support cost" (Interviewee 7, embedded Linux vendor, Europe). "Sharing that effort among many people makes it more likely that you have something good without additional work. It means that you have a support community" (Interviewee 8, embedded Linux vendor, US). "When you need help the community will be there to help you" (Interviewee 9, embedded Linux expert, US).

6 Summary and discussion

This study contributes to the literature on open and distributed innovation and on the changes for firms that their diffusion entails. It employs methodological triangulation—30 in-depth interviews as well as a large-scale survey—to develop a robust and comprehensive picture.

The present paper goes beyond existing work by focusing on the employed OSS programmer as the link between his firm and the OSS community, and on potential principal-agent problems between him and his employer. The latter can arise due to the programmer's double allegiance to his firm and the OSS community, and possible conflicts resulting from it. The fact that, as I show, the developer has a strong say in revealing decisions compounds matters. Furthermore, this and other studies have found that community-related and personal motives do matter for commercial OSS programmers as reasons to reveal code. On the other hand, I have provided evidence that even when acting without management's awareness, OSS programmers may, by revealing code to OSS projects, in fact act in their firm's best interest. The problem may rather lie with management who overestimates the risk and underestimates the benefits of openness. So, how severe is the potential principal-agent problem really?

Considering their attitudes towards OSS, commercial OSS programmers in my sample do, on average, identify with the OSS community and are somewhat ideological about Free Software, but significantly less so than the reference group of non-commercial programmers. As to the stated importance of non-firm-related motives to reveal code, improving one's personal skills matters significantly more for non-commercial programmers. For all other community-related and personal motives I do not find significant differences. In particular, revealing code "because I consider it fair to give back to the community" matters strongly for both groups—a finding well consistent with results from my interviews.

Now, a programmer's fairness towards and identification with the OSS community are not critical per se. Quite the contrary, both are required in order for the firm to capture the potential of OSS.[24] The question is, is the salience of the programmer's "OSS identity" in conflict situations strong enough to override his motivation to abide by the firm's interests? Results obtained by regression analysis are reassuring for firms. I find that the frequency of a

[24] Identification with the OSS community may even be an explicit company goal. Linux distributor Red Hat, e.g., states "Open source forms the foundation of our company's culture" (www.redhat.com/ about/culture/), and it appears plausible that they have to live up to this statement in order to remain successful.

programmer's contributions to public OSS projects is influenced neither by his level of "Free Software ideology," nor by the level of pride he takes in his work. Also the importance of community-related and personal motives to reveal does not show any significant influence. What does exhibit a significant effect, however, is the programmer's level of identification with the OSS community. In light of the findings above, this identification must clearly be distinguished from ideology. As my interviews suggest (see Footnote 23), it will rather be based on positive experiences, in particular on support provided by community members to the respondent's work—which, in turn, is beneficial for his employer (cf. von Hippel, 1987, and Schrader, 1991, for similar findings in the case of information trading). The two identities of a commercial OSS programmer may thus be not so much conflicting, but rather complementing. As Ashforth and Johnson (2001: 48) write somewhat poetically on the issue of conflicting identities, "instead of asking which hat to wear, perhaps we should switch metaphors and ask, say, how a diamond is revealed through its facets."

A limitation of this work is that, while the interviews spanned all hierarchical levels, the survey was directed towards developers. A complementing survey addressing management—or, ideally, various levels in each surveyed firm—would help to identify tensions between developer, middle, and upper management more clearly. A second limitation is that I use respondents' *stated* revealing behavior. Objective information extracted from Internet repositories will likely be difficult to obtain (given that programmers might contribute to different projects under different email addresses) and to match to the survey, but would be superior. Finally, the majority of respondents work for hardware firms (which typically have their hardware as a complementary asset) or for software firms that perform commissioned development. Hence, applicability of my results to other OSS business models (e.g., maintenance) would need to be checked.

This paper points to a number of open questions. With a focus on the role of employed OSS programmers within their firms, my data provide relatively little insight into their role within the respective OSS community. Dahlander and Wallin (2006) address this aspect, focusing, however, on individuals who are explicitly sponsored by their firms to act strategically within the OSS community. Thus, an analysis of the role of the *average* commercial OSS programmer within his community is called for. Second, the dynamics of this role over time and of the programmer's attitudes requires further research, since it determines the sustainability of "commercial" OSS communities. David and Rullani (2006) and Rullani (2006) provide insightful studies of this dynamics, but address mainly non-commercial contributors. Westenholz (2003) describes dynamic identities of OSS programmers between community

and firm, but does neither provide empirical data nor address the principal-agent issue analyzed here. Third, dynamics in firms' attitudes towards openness need to be analyzed. What experiences have firms made with open innovation processes, and what conclusions have they drawn? Does some sort of standard industry practice emerge?

A number of implications for management can be derived. Commercial programmers at the interface between firm and OSS community seem to act responsibly, not as uncontrollable "Free Software ideologists" disregarding their firm's interest. By making suggestions as to what could be revealed, they sometimes act the part of *champions of revealing* as, e.g., in the cases of CEPS and OpenAdaptor. They thus are promoting a process innovation, namely, the active use of and participation in public OSS development. While acts of excessive revealing probably do occur sometimes, an overly restrictive attitude by firm managers and lawyers might in fact be more damaging than helpful, hindering a promising new form of development collaboration. This is all the more true since cases of grave misconduct can most likely be proven by analyzing mailing lists and code repositories. The importance of OSS developers at the interface between firm and community is particularly high when, due to the extent of their activity, they qualify as boundary spanners (Allen, 1977; Tushman, 1977). As such, they stimulate the innovation process by bridging organizational, and thus often also technical, boundaries.[25]

A specific recommendation for human resource management is to screen out, when hiring OSS developers, those with an overly strong ideological attitude towards the "freedom of software." In contrast, a job candidate's strong identification with the OSS community actually appears desirable for the prospective employer.

Tentative normative implications thus are not so much to monitor developers' revealing behavior more strictly, but rather to soften the stance towards intellectual property protection. At the same time, the firm must provide clear policies in this regard, and make sure it does not attract and hire, as an "open" firm, avid supporters of "Free Software ideology."

Open and distributed innovation processes as observed in the field of OSS carry a high potential. In order to take full advantage of it, firms need to strike the right balance between

---

[25] See Henderson and Clark (1990). For the specific case of technical boundaries between commercial firms focusing on different technical fields within OSS development, see Henkel (2004b). For boundary spanners in open innovation communities, see Fleming and Waguespack (2007).

protection and openness.[26] In particular, this requires rethinking engrained practices of overly tight intellectual property management: a more positive attitude towards revealing will enable firms to better share in the benefits of open innovation processes.

[26] A similar statement was made by I.B.M.'s senior vice president John E. Kelly: "Every action we take will be done with an eye toward striking this more subtle balance between proprietary and open" (New York Times, "I.B.M. Hopes to Profit by Making Patents Available Free," April 11, 2005).

Appendix

Table 1: Firm policies towards revealing (commercial developers, N = 197)

Statement	% who agreed	Cross table: % who agreed to the 'line' and the 'column' statement			
		1.	2.	3.	4.
1. "There is no official policy"	46.7%				
2. "I make suggestions as to what we could make public, and discuss each case with my supervisor"	34.0%	13.7%			
3. "It is in my responsibility to reveal source code"	25.4%	12.2%	9.6%		
4. "My company encourages me to contribute source code that is not critical for competition"	22.8%	4.6%	9.1%	8.6%	
5. "My company is very restrictive in revealing code"	16.8%	4.1%	5.6%	2.0%	0.5%

Table 2: Attitudes towards OSS, by type of developer

Statement	Type of respondent	disagree strongly (-2)	disagree somewhat (-1)	neutral (0)	agree somewhat (1)	agree strongly (2)	N	median	mean	Mann-Whitney test (p)
OSSidentification: "I identify with the open source community"	com-mercial	1.6%	3.8%	13.1%	30.6%	50.8%	183	2	1.25	0.061
	non-com-mercial	1.5%	0.0%	6.2%	30.8%	61.5%	65	2	1.51	
FreeSWideology: "Free softwarematters because all freedom matters"	com-mercial	6.0%	6.6%	23.0%	33.9%	30.6%	183	1	0.77	0.003
	non-com-mercial	0.0%	6.1%	15.2%	28.8%	50.0%	66	1.5	1.23	
Pride: "I am really proud of my work developing open source software and	com-mercial	2.2%	2.8%	19.0%	34.1%	41.9%	179	1	1.11	0.105
	non-com-	0.0%	3.2%	12.7%	31.7%	52.4%	63	2	1.33	

embedded Linux"	mercial									

Table 3: Community-related and personal reasons to reveal, by type of developer

Type of respondent	disagree strongly (–2)	disgree somewhat (–1)	neither agree nor disagree (0)	agree somewhat (+1)	agree strongly (+2)	N	median	mean	Mann-Whitney test (p-value)
I reveal code because…									
... I consider it fair to give back to the community.									
commercial	0.0%	1.1%	3.9%	28.9%	66.1%	180	2	1.60	0.647
non-com.	0.0%	0.0%	3.1%	28.1%	68.8%	64	2	1.66	
... in the long run, you only get something when you gave something before									
commercial	2.3%	9.0%	22.6%	29.9%	36.2%	177	1	0.89	0.754
non-com.	3.2%	4.8%	25.4%	28.6%	38.1%	63	1	0.94	
... I get feedback on my code, which improves my personal skills									
commercial	1.7%	6.3%	21.6%	41.5%	29.0%	176	1	0.90	0.028
non-com.	1.6%	6.4%	12.7%	33.3%	46.0%	63	1	1.16	
... I demonstrate my skills to future employers									
commercial	6.4%	17.9%	34.1%	24.9%	16.8%	173	0	0.28	0.238
non-com.	9.7%	11.3%	24.2%	35.5%	19.4%	62	1	0.44	

Table 4: Frequency of code contributions to public embedded Linux projects

Developers working for …	never (1)	once a year (2)	once a month (3)	once a week (4)	several times / week (5)	N
Software firms	15.4%	18.0%	46.2%	12.8%	7.7%	39
Device manufacturers	27.3%	34.9%	27.3%	3.0%	7.6%	66
Component manufacturers	41.7%	16.7%	33.3%	0.0%	8.3%	12
Total, commercial developers	24.8%	27.4%	34.2%	6.0%	7.7%	117

Note: Only those observations reported which are used in the regression analysis

Table 5: Descriptive statistics of explanatory variables not covered in Table 2

	Dummy variable, equal to "1" if …	frequency of "1"	
SizeMedium	firm has 11 to 200 employees	47	(40.2%)
SizeLarge	firm has more than 200 employees	35	(29.9%)
TypeDeviceMfr	firm is a device manufacturer	66	(56.4%)
TypeComponentMfr	firm is a component manufacturer	12	(10.3%)
PolEncouraging	firm has encouraging policy towards revealing OSS code	31	(26.5%)
PolRestrictive	firm has restrictive policy towards revealing OSS code	14	(12.0%)

		no	same time	yes
OSSbeforeEmbedded	respondent has worked on OSS longer than on embedded software	68 (58.1%)	13 (11.1%)	36 (30.8%)

	Mean	Std. Dev.	Median	Min	Max
DevelYearsEL (yrs)	2.78	1.69	3.0	0.2	10.0
ELTimePerWeek (hrs)	34.67	21.71	30.0	1	110
ShareSpareTime	16.7%	18.5%	11.0%	0.0%	66.7%
CompanyYearsEL (yrs)	3.68	1.93	4	0.0	10.0
ReasonsCommunity (–2 … +2)	1.27	0.72	1.5	-1	2
ReasonsPersBenef (–2 … +2)	0.64	0.87	0.75	-2.0	2.0

Note: Only those observations reported which are used in the regression analysis (N = 117)

Table 6: Correlation between explanatory variables

	Pol-Encou-raging	Pol-Restric-tive	OSS-identi-fication	FreeSW-ideology	Pride	Devel-YearsEL	ELTime-PerWeek	Share-Spare-Time	OSS-before-Embed.	Size-Medium	Size-Large	Com-pany-YearsEL	Type-Device-Man	Type-Compo-nentMfr	Reasons-Com-munity
PolRe-strictive	-0.2213 (0.0165)	1.0000													
OSSiden-tification			1.0000												
FreeSW-ideology			0.3434 (0.0002)	1.0000											
Pride			0.4309 (0.0000)	0.2347 (0.0109)	1.0000										
Devel-YearsEL	0.2837 (0.0019)		0.3385 (0.0002)		0.1858 (0.0449)	1.0000									
ELTime-PerWeek			0.1629 (0.0792)	-0.1716 (0.0644)	0.2616 (0.0044)	0.2073 (0.0249)	1.0000								
Share-SpareTime			0.1994 (0.0311)	0.1871 (0.0434)	0.1823 (0.0491)	0.1581 (0.0887)		1.0000							
OSSbe-foreEmbed			0.2220 (0.0162)		0.3707 (0.0000)				1.0000						
Size-Medium				-0.2074 (0.0248)						1.0000					
Size-Large		0.1617 (0.0815)								mutually exclusive	1.0000				
Company-YearsEL	0.2602 (0.0046)					0.4691 (0.0000)	0.1862 (0.0445)	0.1573 (0.0902)				1.0000			
TypeDevice-Mfr						-0.2000 (0.0306)	-0.2955 (0.0012)						1.0000		
TypeCom-ponentMfr					0.2175 (0.0185)						0.3329 (0.0002)		mutually exclusive	1.0000	
Reasons-Community			0.2725 (0.0030)	0.2780 (0.0024)	0.2827 (0.0020)	0.1845 (0.0464)		0.2171 (0.0187)							1.0000
Reasons-PersBenef			0.2844 (0.0019)	0.3200 (0.0004)	0.4244 (0.0000)	0.1756 (0.0582)	0.2176 (0.0184)	0.2305 (0.0124)						0.1728 (0.0625)	0.4024 (0.0000)

Spearman rank correlation (rho), p-value in parentheses. Correlation coefficients printed for $p \leq 0.1$. N = 117

Table 7: Ordered Probit regressions of the frequency of code contributions (1)

	spec 1a	spec 1b	marg. eff. 1	marg. eff. 2	marg. eff. 3	marg. eff. 4	marg. eff. 5	x
Explanatory variables related to Hypotheses H3 to H8								
PolSupportiveIndex	0.575*** (0.182)	0.596*** (0.175)	-0.143*** (0.047)	-0.094** (0.037)	0.168*** (0.057)	0.046** (0.021)	0.024* (0.014)	0.145
OSSidentification	0.333*** (0.123)	0.267** (0.108)	-0.064** (0.026)	-0.042** (0.022)	0.0753** (0.033)	0.021* (0.011)	0.011* (0.006)	1.350
FreeSWideology	-0.025 (0.096)							
Pride	-0.135 (0.144)							
Reasons-Community	0.073 (0.166)							
Reasons-PersBenef	0.012 (0.147)							
Control variables related to the developer								
DevelYearsEL	0.082 (0.062)	0.104* (0.056)	-0.025* (0.014)	-0.016* (0.010)	0.029* (0.016)	0.008 (0.005)	0.004 (0.003)	2.775
ELTotalTime	0.023*** (0.005)	0.024*** (0.005)	-0.006*** (0.001)	-0.004*** (0.001)	0.007*** (0.002)	0.002*** (0.001)	0.001** (0.001)	34.67
ShareSpareTime	1.721*** (0.627)	1.809*** (0.572)	-0.436*** (0.152)	-0.286** (0.117)	0.509*** (0.175)	0.139* (0.072)	0.073* (0.039)	0.168
OSSbefore-Embed	0.359*** (0.128)	0.320*** (0.122)	-0.077*** (0.030)	-0.050** (0.025)	0.090** (0.037)	0.025* (0.014)	0.013* (0.007)	-0.274
Control variables related to the firm								
SizeMedium	-0.049 (0.254)							
SizeLarge	-0.449 (0.319)	-0.409 (0.250)	0.107 (0.071)	0.054* (0.032)	-0.119 (0.079)	-0.028* (0.016)	-0.014 (0.009)	0.300
CompanyYearsEL	0.056 (0.059)							
TypeDeviceMfr	-0.072 (0.231)							
TypeComponentMfr	-0.132 (0.517)							
Wald test	$\chi^2(15)=75.1$ p<0.0001	$\chi^2(7)=70.5$ p<0.0001						
Pseudo R-squared	0.227	0.219						

Robust standard errors in parentheses. Significance levels 10% (*), 5% (**), 1% (***). Marginal effect (n) relates to the n'th category of contribution behavior, see Table 4. x: value of the respective variable at which marginal effects are calculated. N = 117 in all specifications.

Table 8: Ordered Probit regressions of the frequency of code contributions (2)

	spec 2a	spec 2b	spec 3a	spec 3b
Explanatory variables related to Hypotheses H3 to H9				
PolEncouraging	0.574**	0.624**		
	-0.247)	(0.244)		
PolRestrictive	-0.577	-0.548		
	(0.365)	(0.348)		
OSSidentification	0.333***	0.269**	0.283**	0.223**
	(0.122)	(0.109)	(0.116)	(0.111)
FreeSWideology	-0.025		-0.044	
	(0.095)		(0.097)	
Pride	-0.135		-0.173	
	(0.143)		(0.144)	
Reasons-Community	0.073		0.14	
	(0.163)		(0.166)	
Reasons-PersBenef	0.012		0.091	
	(0.147)		(0.146)	
Control variables related to the developer				
DevelYearsEL	0.082	0.103*	0.123*	0.151***
	(0.062)	(0.057)	(0.063)	(0.057)
ELTotalTime	0.023***	0.024***	0.022***	0.022***
	(0.005)	(0.005)	(0.005)	(0.005)
ShareSpareTime	1.721***	1.796***	1.562***	1.732***
	(0.626)	(0.574)	(0.602)	(0.540)
OSSbeforeEmbed	0.359***	0.321***	0.338***	0.292**
	(0.129)	(0.123)	(0.129)	(0.121)
Control variables related to the firm				
SizeMedium	-0.048		-0.108	
	(0.253)		(0.256)	
SizeLarge	-0.449	-0.411	-0.560*	-0.508**
	(0.318)	(0.251)	(0.312)	(0.239)
CompanyYearsEL	0.056		0.054	
	(0.061)		(0.063)	
TypeDeviceMfr	-0.072		-0.1	
	(0.232)		(0.238)	
TypeComponentMfr	-0.133		-0.25	
	(0.518)		(0.492)	
Wald test	$\chi^2(16) = 75.4$ $p < 0.0001$	$\chi^2(8) = 70.1$ $p < 0.0001$	$\chi^2(14) = 71.7$ $p < 0.0001$	$\chi^2(6) = 65.1$ $p < 0.0001$
Pseudo R-squared	0.227	0.219	0.200	0.188

Robust standard errors in parentheses. Significance levels 10% (*), 5% (**), 1% (***). N = 117.

References

Alexy, O. and M. Leitner (2008), 'Norms, rewards, and their effect on the motivation of open source software developers,' *Technical University of Munich working paper*, http://ssrn.com/abstract=1007689.

Allen, T. J. (1977), *Managing the Flow of Technology*. MIT Press: Cambridge, MA.

Allen, T. J. (1988), 'Distinguishing engineers from scientists,' R. Katz (ed.), *Managing Professionals in Innovative Organizations*. Harper Business: New York. 5-18.

Arora, A., A. Fosfuri and A. Gambardella (2001), *Markets for Technology: The Economics of Innovation and Corporate Strategy*. MIT Press: Cambridge, MA.

Ashforth, B. E. (2001), *Role Transitions in Organizational Life: An identity-based Perspective*. Erlbaum: Mahwah, NJ.

Ashforth, B. E. and S. A. Johnson (2001), 'Which hat to wear: The relative salience of multiple identities in organizational contexts,' in M. A. Hogg and D. J. Terry (eds.), *Social Identity Processes in Organizational Contexts*. Psychology Press: Philadelphia, PA. 31-48.

Behlendorf, B. (1999), 'Open source as a business strategy,' C. Dibona, S. Ockman and M. Stone (eds.), *Open-Sources: Voices from the Open Source Revolution*. O'Reilly: Sebastopol, CA. 149-170.

Bonaccorsi, A., S. Giannangeli and C. Rossi (2006), 'Entry strategies under competing standards: Hybrid business models in the open source software industry,' *Management Science*, **52**(7), 1085-1098.

Bonaccorsi, A. and C. Rossi (2006), 'Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business,' *Knowledge, Technology and Policy*, **18**(4), 40-64.

Brown, J. S. and P. Duguid (1991), 'Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation,' *Organization Science,* **2**(1), 40-57.

Chesbrough, H. (2003), *Open Innovation: The New Imperative for Creating and Profiting from Technology*. Harvard Business School Press: Boston, MA.

Dahl, M. S. and C. O. R. Pedersen (2004), 'Knowledge flows through informal contacts in industrial clusters: myth or reality?' *Research Policy,* **33**(10), 1673-1686.

Dahlander, L. (2004), 'Appropriating the commons: Firms in open source software,' Chalmers University of Technology, Gothenburg, Working Paper, http://opensource.mit.edu/papers/dahlander2.pdf.

Dahlander, L. (2007), '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.

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

Dahlander, L. and M. W. Wallin (2006), 'A man on the inside: unlocking communities as complementary assets,' *Research Policy*, **35**, 1243–1259.

David, P.A. and F. Rullani (2006), 'The micro-dynamics of open source software development activity,' in E. Damiani, B. Fitzgerald, W. Scacchi, M. Scotto, and G. Succi (eds.), IFIP Intemational Federation for Information Processing, Volume 203, *Open Source Systems*. Springer: Boston, MA, 339-340.

Engelfriet, A. (2006), 'Best of both worlds: Real-world examples of mixing open and closed source software,' *Presentation at Open Source Business Conference* (OSBC), San Franciso.

Feller, J. and B. Fitzgerald (2002), *Understanding Open Source Software Development*. Addison Wesley: Boston, MA.

Festinger, L. A. (1957), *A Theory of Cognitive Dissonance*. Stanford University Press: Stanford.

Fleming, L. and D. M. Waguespack (2007), 'Brokerage, boundary sanning, and leadership in open innovation communities,' *Organization Science*, **18**(2), 165–180.

Fosfuri, A., M. S. Giarratana and A. Luzzi (2008), 'The Penguin Has Entered the Building: The Commercialization of Open Source Software Products,' *Organization Science*, forthcoming.

Franck, E. and C. Jungwirth (2003), 'Reconciling investors and donators – the governance structure of open source,' *Journal of Management and Governance*, **7**, 401-421.

Franke, N. and E. von Hippel (2003), 'Satisfying heterogeneous user needs via innovation toolkits: The case of Apache security software,' *Research Policy*, **32**(7), 1199-1215.

Gassmann, O. (2006), 'Opening up the innovation process: Towards an agenda,' *R&D Management*, **36**, 223-226.

Ghosh, R. A., R. Glott, B. Krieger and G. Robles (2002), 'Free/libre and open source software: Survey and study. Part 4: Survey of developers,' International Institute of Infonomics, University of Maastricht, http://www.infonomics.nl/FLOSS/report/.

Gruber, M. and J. Henkel (2005), 'New ventures based on open innovation—an empirical analysis of start-up firms in embedded Linux,' *International Journal of Technology Management,* **33**(4), 354-372.

Hamel, G., Y. L. Doz and C. K. Prahalad (1989), 'Collaborate with your competitors – and win,' *Harvard Business Review,* **67**(1), 133-139.

Hars, A. and S. Ou (2002), 'Working for free? Motivations for participating in open-source projects,' *International Journal of Electronic Commerce,* **6**(3), 25–39.

Hecker, F. (1999), 'Setting up shop: The business of open-source software,' *IEEE Software,* **16**(1), 45-51.

Henderson, R. M., K. B. Clark (1990), 'Architectural innovation: the reconfiguration of existing product technologies and the failure of established firms,' *Administrative Science Quarterly*, **35**(1), 9-16.

Henkel, J. (2004a), 'Open source software from commercial firms—tools, complements, and collective invention,' *Zeitschrift für Betriebswirtschaft*, Supplement 4, 1-23, www.tim.wi.tum.de/home/index.php?option=com_docman&task=doc_download&gid=56&Itemid=90.

Henkel, J. (2004b), 'The Jukebox Mode of innovation—a model of commercial open source development,' *CEPR Discussion Paper* 4507. http://ssrn.com/abstract=578142.

Henkel, J. (2006), 'Selective revealing in open innovation processes: The case of embedded Linux,' *Research Policy,* **35**(7), 965-969.

Hertel, G. and S. Niedner, S. Herrmann (2003), 'Motivation of software developers in open source projects: An internet-based survey of contributors to the Linux kernel,' *Research Policy,* **32**(7), 1159-1177.

Howell, J. M. and C. A. Higgins (1990), 'Champions of technological innovation,' *Administrative Science Quarterly,* **35**(2), 317-341.

Jaaksi, A. (2006), 'Building consumer products with open source communities – the Maemo and 770 experiences,' *Presentation at LinuxWorld* 2006, Boston, http://www.kotiposti.net/ jaaksi/ME9_LinuxWorld_2006_AriJaaksi_.pdf.

Klandermans, B. (1997), *The Social Psychology of Protest*. Basil Blackwell: Oxford, UK.

Lakhani, K. R. and R. G. Wolf (2005), 'Why hackers do what they do: Understanding motivation and effort in free/open source software projects,' in J. Feller, B. Fitzgerald, S. Hissam, and K. R. Lakhani (eds.), *Perspectives on Free and Open Source Software*. MIT Press: Cambridge, MA.

Lee, G. K. and R. E. Cole (2003), 'From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development,' *Organization Science,* **14**(6), 633-649.

Lee, S., N. Moisa and M. Weiss (2003), 'Open source as a signalling device – An economic analysis,' in Feller, J., B. Fitzgerald, S. Hissam and K. Lakhani (eds.), *Taking Stock of the Bazaar: Proceedings of the 3rd Workshop on Open Source Software Engineering,* http://opensource.ucc.ie/icse2003.

Lerner, J. and J. Tirole (2002), 'Some simple economics of open source,' *Journal of Industrial Economics,* **50**(2), 197–234.

Osterloh, M. and S. G. Rota (2007), 'Open source software development – Just another case of collective invention?,' Research Policy, **36**(2), 157-171

Raymond, E. S. (1999), *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly: Sebastopol, CA.

Rullani, F. (2006), 'Dragging developers towards the core: How the free/libre/open source software community enhances developers' contribution,' Sant'Anna School of Advanced Studies, Pisa, Working paper.

Schrader, S. (1991), 'Informal technology transfer between firms: Cooperation through information trading,' *Research Policy,* **20**(2), 153–170.

Simon, B., M. Loewy, S. Stürmer, U. Weber, P. Freytag, C. Habig, C. Kampmeier, P. Spahlinger (1998), 'Collective identification and social movement participation,' *Journal of Personality and Social Psychology*, **74**, 646–658.

Stallman, R. (1999), 'The GNU operating system and the free software movement,' in DiBona, C., S. Ockman and M. Stone (eds.), *Opensources: Voices from the Open Source Revolution*. O'Reilly: Sebastopol, CA. 53–70.

Strauss, A. L. and J. M. Corbin (1990), *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. Sage: Beverly Hills, CA.

Stryker, S. (1980), *Symbolic Interactionism: A Social Structural Version*. Benjamin/Cummings: Menlo Park, CA.

Teece, D. J. (1986), 'Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy,' *Research Policy,* **15**(6), 285–305.

Torvalds, L. and D. Diamond (2001), *Just for Fun: The Story of an Accidental Revolutionary*. HarperCollins.

Tushman, M. (1977), 'Special boundary roles in the innovation process,' *Administration Science Quarterly*, **22**, 587-605.

VDC (2004), White paper, Venture Development Corporation. Referenced in: Linux now top choice of embedded developers, www.linuxdevices.com/news/NS2744182736.html.

von Hippel, E. (1987), 'Cooperation between rivals: Informal know-how trading,' *Research Policy,* **16**(6), 291–302.

von Hippel, E. (2001), 'Innovation by user communities: Learning from open source software,' *MIT Sloan Management Review* 2001(Summer), 82–86.

von Hippel, E. and G. von Krogh (2003), 'Open source software and the "Private-Collective" innovation model: Issues for organization science,' *Organization Science,* **14**(2), 209–223.

Webb, W. (2002), 'Pick and place: Linux grabs the embedded market,' www.reed-electronics.com/ednmag/contents/images/253780.pdf.

Webb, W. (2006), 'Linux joins the consumer-electronics revolution,' *edn.com*, February 16, 2006, http://www.edn.com/contents/images/6305349.pdf.

West, J. (2003), 'How open is open enough? Melding proprietary and open source platform strategies,' *Research Policy,* **32**(7), 1259-1285.

West, J. and Gallagher, S. (2006), 'Challenges of open innovation: The paradox of firm investment in open-source software,' *R&D Management*, **36(3)**, 319-331.

Westenholz, A. (2003), 'Identity work in the fractures between open source communities and the economic world,' *Copenhagen Business School working paper* 2003.16, http://ir.lib.cbs.dk/download/ISBN/x656378483.pdf.

Wichmann, T. (2002), 'FLOSS final report – part 2: Free/libre open source software: Survey and study – firms' open source activities: Motivations and policy implications,' *Berlecon Research,* http://www.berlecon.de/studien/downloads/200207FLOSS Activities.pdf.