ETLA
ELINKEINOELÄMÄN TUTKIMUSLAITOS
THE RESEARCH INSTITUTE OF THE FINNISH ECONOMY
Lönnrotinkatu 4 B 00120 Helsinki Finland Tel. 358-9-609 900
Telefax 358-9-601 753 World Wide Web: http://www.etla.fi/

Keskusteluaiheita – Discussion papers

No. 947

Markku Stenborg*

EXPLAINING OPEN SOURCE

* Contact information: Markku Stenborg, ETLA – The Research Institute of the Finnish Economy, Lönnrotinkatu 4 B, FIN-00120 Helsinki, Finland, mobile +358 50 577 6340, fax +358 9 601 753, <mailto:markku.stenborg@etla.fi>.
Financial support from The National Technology Agency (Tekes) is gratefully acknowledged. Comments by Pekka Ylä-Anttila and Heli Koski are gratefully acknowledged. The usual disclaimer applies.

ISSN 0781-6847 11.11.2004

STENBORG, Markku, **EXPLAINING OPEN SOURCE.** Helsinki: ETLA, Elinkeinoelämän Tutkimuslaitos, The Researsch Institute of the Finnish Economy, 2004, 34 p. (Keskusteluaiheita, Discussion Papers, ISSN 0781-6847, no. 947).

ABSTRACT: This paper looks at stories and evidence set forth to explain programmer participation in open source projects, coordination of these projects, licensing of open source software, and on the public policy regarding open source software. The paper concentrates on economics and empirical evidence.

JEL classification: L86, L31, H41,

Keywords: Open Source Software, Incentives, Licensing Choice, Coordination, Empirical Evidence

STENBORG, Markku, **EXPLAINING OPEN SOURCE.** Helsinki: ETLA, Elinkeinoelämän Tutkimuslaitos, The Researsch Institute of the Finnish Economy, 2004, 34 s. (Keskusteluaiheita, Discussion Papers, ISSN 0781-6847, no. 947).

TIIVISTELMÄ: Tutkimuksessa tarkastellaan tarinoita ja evidenssiä, joilla selitetään ohjelmoijien osallistumista avoimen lähdekoodin projekteihin, kyseisten projektien hallinnointia, avoimen lähdekoodin ohjelmistojen lisensointia sekä politiikkasuosituksia. Tutkimus keskittyy ilmiön taloustieteelliseen tarkasteluun sekä empiiriseen evidenssiin.

1 Introduction

Currently, there exist thousands of open source software projects, ranging from small utilities and device drivers to large and complex systems such as Linux, Open Office, Apache and MySQL. Open source has proved to be a successful mode of innovation and software production. Several open source projects have captured market shares from their commercial rivals and have a reputation for innovation and reliability. For instance, GNU/Linux and other free UNIX-based operating systems are the only real challengers to the Microsoft Windows operating system for Intel-based PCs, Sendmail routes at least 42% of mails in the Internet, and Apache dominates the web server market. Further, arguably, some of the open source products are of better quality than competing commercial products.[1]

Since the early days of computing, user/programmers have shared computer code. Many important early programs, including many developed with government funding, were widely shared. The mode of software development was similar to that of science (see, e.g., in Dasgupta and David, 1994). In the 1950's and 60's, proprietary software consisted of limited applications that were almost entirely sold bundled with computer hardware. Packaged software was rarely sold until the 1970's, when IBM was challenged by private and government lawsuits to unbundle software and hardware, and when mini-computers became widely used (Parker and Grimm, 2000). In the mid-1980, a new, more formalized form of sharing software code emerged. Richard Stallman, concerned about his ability to access, modify and improve software, started the Free Software movement and developed the GNU Public License (GPL) for software.[2] Under the GPL, the user obtains free access to the software code and agrees that any redistribution of the code will also be freely available, including any modifications the user makes to the code (this is the often-mentioned "viral" characteristic of the license).

The Free Software movement gained momentum during the mid-90's, with the rise of the Internet. Developers such as Linus Torvalds, the initial creator of Linux, pioneered new methods of development that permitted hundreds of volunteer programmers to participate in joint software development over the Internet. Out of this broad participation arose the open source movement, which includes software developed under the GPL as well as other license agreements. The Linux project is one of

[1] See, e.g., Wheeler (2004) and the references therein.

[2] For more on the history of the open source movement, see, e.g., Moody (2001) and Weber (2004), Chapters 2 and 4.

the most successful open source projects to date. In 2003, Linux held 23.1 % share of the server operating system market (Gonsalves, 2003), while in 1995, it accounted for less than half of 1 % of this market (Di Carlo, 2002).[3]

The open source phenomenon has attracted increased scientific attention recently, not only from software engineers but also from economists and other social scientists. In his book on politics and economics of open source, Steven Weber (2004) summarizes the need for research in three broad areas: the motivation of individuals or the microeconomic foundations of open source, the coordination of effort, and the complexity of software. In addition, fourth and fifth economics-related research topics are the licensing choices for the open source software, and public policy and business strategy implications from adopting open source, either as a user or as a development mode for a software producer, respectively.

To introduce these research topics, note first that the open source process depends on individual behavior that is rather surprising. Public goods theory predicts that nonrival and non-excludable goods, such as open source software, ought to encourage free riding. Why, then, do programmers choose voluntarily to allocate some or a substantial portion of their time and effort to a joint project for which they will not be compensated? Second, how does the open source process sustain coordinated cooperation among a large number of contributors, outside the bounds of hierarchical or market mechanisms? Any production process depends on pulling together individual efforts in a way that they add up to a functioning product. Authority within a firm and the price mechanism across agents are standard means to coordinate specialized knowledge and highly differentiated division of labor, but neither is operative in open source. Instead, individuals choose for themselves what they want to work on.

Third, software is a complex technical artifact. In *The Mythical Man-Month*, a classic study of the organization of computer programming, Frederick Brooks noted that when large organizations add manpower to a software project that is behind schedule, the project typically falls even further behind schedule. He explained this with an argument that is now known as Brooks' Law: as you raise the number of programmers on a project, the work that gets done scales linearly, while complexity and vulnerability to mistakes scales geometrically. This is supposed to be inherent in the logic of the division of labor – the geometric progression represents the scaling of the

[3] However, currently there exists very limited research on the adoption and diffusion of open source software.

number of possible communication paths and interfaces between pieces of code written by individual developers.[4]

This paper provides a selective survey of contributions to the understanding of open source; the bias here is economics and empirical evidence. There exists several case studies of open source projects, some of which will discussed very briefly below, but there are only a handful of econometric papers. Krishnamurthy (2002) examined 100 mature products available on the SourceForge web site, and found that most of the products have very few developers and that most open source software programs do not generate a great deal of discussion. Lerner and Tirole (2005) examined the choice of licenses using an approximately 39,000 open source projects from the SourceForge web site. They find that projects that run on commercial operating systems and projects that are designed for developers tend to use less restrictive licenses, while projects that are targeted for end users tend to use more restrictive licenses. Bonaccorsi and Rossi (2003a, b, c) conducted surveys on Italian firms, and found that firms that employ restrictive licenses supply (on average) fewer proprietary products than firms that do not and firms that use restrictive licenses attach greater importance to social motivation. Fershtman and Gandel (2004) use a data set consisting of 71 open source projects hosted at the SourceForge web site, covering an eighteen-month period, with data collected at two-month intervals. They found that that the output per contributor of open source programs is much higher when licenses are less restrictive. Hann et al. (2004) exploited the fact that the Apache Software Foundation ranks its members based on merit, and found that greater open source experience, as measured in contributions made, does not result in wage increases for contributors, while achieving a higher status in was associated with a 13-27% increase in wages, depending on the rank attained.[5] These papers are discussed in more detail below. But to better understand the empirical findings, we also need to explore the theoretical contributions of the open source issues.

The paper is organized as follows. In the next Section, I review various hypotheses set forth to explain the public goods paradox. In Section 3, I will focus on the governance and coordination of open source projects. In Section 4, I will look at licensing issues. In section 5, I briefly touch some of the issues related to public policy. Section 6 has some concluding comments.

---

[4] Weber (2004) provides a thorough account on these issues.

[5] See also the FLOSS survey, in which Ghosh et al. (2002) conducted an online survey on 2 784 open source developers. They found, for instance, that an initial motivation for participation in open source community is to aim at individual skills and to the exchange of information and knowledge with other developers.

2 Motives to Contribute to Open Source

Open source software is a public good. Anyone can download a copy of Linux, say, along with its source code for free, which means it is non-excludable. As it is a digital product that can be replicated at no cost, it is non-rival. Open source software relies on volunteering for the provision of new code, bug fixes of the existing code, and online help with problems running and installing the program. According to the established economic theory, private provision of a public good should be Pareto inefficient, as individuals face an incentive to free ride on the contributions of the others. Second, large scale open source systems such as Linux are subject to collective provision constraint, as their production depends on coordinated contributions from a large number of developers. Many programming tasks are often best performed by a single individual. Then one person bears the entire cost of providing a software module which benefits the whole community. Individuals then have strong incentives to free ride and let someone else take on the job. Third, a key element here is voluntary selection of tasks, as each person is free to choose what she wishes to work on or to contribute. This brings about the problem of coordinating individual effort, discussed in more detail in the next Section. Smith and Kollock (1999) have even called Linux the impossible public good. In theory, then, we should not see much open source software, and especially large scale open source systems such as Linux should be unsuccessful attempts against economic logic.

Perhaps the most fundamental question seeking for an answer is, what are the incentives of these programmers to invest time and effort in developing open source programs? This topic has received the most attention from economists and other social scientists, and some explanations for the motivation paradox have already been proposed. Many social scientist view that one should concentrate on non-monetary incentives such as personal satisfaction, ideology, and professional status. Many view that open source process has incentives similar to those in academia. In general, individual motivations can be grouped under two broad headings, intrinsic and extrinsic (Hars and Ou, 2001; Lakhani and Wolf, 2003). Intrinsic motivation is valued *per se*,[6] and these motivations are discussed next in Section 2.1. Extrinsic motivations relate

---

[6] "Intrinsic motivation is defined as the doing of an activity for its inherent satisfactions rather than for some separable consequence. When intrinsically motivated, a person is moved to act for the fun or challenge entailed rather than because of external prods, pressures or rewards" (Ryan and Deci, 2000). Components of intrinsic motivation are self-determination and autonomy that arise from individual's perception of freedom, identity, responsibility, and control and competence, which is positively related to intrinsic motivation to the extent that the task is neither far below nor far above the individual's level of competence (Deci, 1980; Deci and Ryan, 1985).

to the immediate or delayed benefits to the individual, typically through monetary compensation, and these will be discussed in the Section 2.2.

2.1 Altruism, Free Speech, etc.: The Idealism of Open Source

Based on an online survey on 2 784 open source developers, Ghosh et al. (2002) report insights in the open source community. The community is a rather young and dominantly male, with a professional background in the IT sector and a high educational level. They are mostly singles with a high degree of mobility. Developing open source resembles a hobby rather than salaried work. Engineers, programmers and students play a significant role in the community, but project performance and leadership is primarily a matter of professionals. An initial motivation for participation in the open source development aims at individual skills and exchange of information and knowledge with other developers.

Many open source advocates have portrayed the movement to be outside of the economic rationalism, and emphasized individual enjoyment of programming, altruism, gift-giving, sharing, reciprocity, identification with the community, and other such social and moral virtues as explanands for programmer behavior. Richard Stallman (e.g., Stallman 1999) has set forth normative arguments about the nature of software as scientific knowledge, not a proprietary product, thus something to be shared and distributed "like sharing of recipes among cooks". Eric Raymond (1998a,b, 1999) and other developers offer ego gratification as an explanation, based on, inter alia, observations that hackers are motivationally very much like artists, in the sense that they seek fun, challenge, and beauty in their work.[7]

Also the "hacker identity" seems to be very important in surveys. Intrinsic motivations have been suggested to arise from identification with the open source community. Eric Raymond depicts the open source community as a gift-giving culture in which economic scarcity is not an issue and social status is determined "not by what you control but by what you give away" (Raymond, 1999). There is a set of core norms and beliefs that guides the open source community: software should be free; there is high value attached to the sharing of information; technical knowledge and high-quality source code are appreciated; reputation is earned among peers; and recognizing a developer's contribution by giving it appropriate credit is important

[7] For more on political economics of motivation to provide open source, see, e.g., Weber (2004), Chapters 4 and 5.

(Weber, 2004, Chapter 3).[8] Bergquist and Ljungberg (2001) make a parallel with peer review in academic communities, with status and reputation provided by peers.[9] Interactions in open source communities are anonymous, which obviously prevent direct paybacks, but still it is thought that voluntary contributions can be supported by an expectation of generalized reciprocity (See, e.g., Kollock, 1999, among others).

The sense of identification with the community as a motivation for participation to open source projects has been empirical analyzed by social scientists.[10] For instance, in their study, Hertel et al. (2003) found that identification with the community, improving own software and tolerance of time investments had the strongest influence on participants engagement in the Linux project. They also found that the perceived importance of one's contribution for the group outcome, the subjective evaluation of team goals and the willingness to engage in programming in the future were the best predictors for investment of time, and that perceived self-efficacy is the best predictor for performance (approximated by number of lines of code and patches contributed). In their survey, Hars and Ou (2001) found that altruism, feeling of accomplishment, competence, enjoyment and effectiveness were highly rated motivators. Lakhani and Wolf (2003) report that the sense of creativity and intellectual stimulation were the strongest and most pervasive factors in explaining participation to open source.

But if the motivation to take part in an open source project is professional recognition or status, then we need to examine status concerns as an incentive. The general idea is that status is a relative ranking of individuals and as such it cannot be a perfect substitute for monetary rewards. Then, giving status to one individual implies lower status to others. Such a characterization may imply tournament-like incentives, but it also implies that the use of status as professional incentives will be limited. If a society gives too many medals, say, then the value of a medal diminishes.[11] It is not clear that programmers will continue to be motivated by status when the value of status diminishes.

---

[8] See, e.g., Edwards (2001) and Cohendet, Creplet and Dupouet (2001). This story to explain the motivation of open source developers is based on the theories of epistemic communities and situated learning.

[9] See also Bezroukov (1999), Raymond (1999c), Kelty (2001) and David et al. (2001).

[10] See also Gosh et al. (2002) and Stewart and Gosain (2003).

[11] For some economics of status, see, e.g., Fershtman and Weiss (1993), Fershtman, Murphy and Weiss (1996), and Auriol and Renault (2003).

2.2 Reputation, Signaling and Such: Economics Strikes Back

Ghosh et al. (2002) point out that about a third of the developers surveyed are being paid directly for developing open source software. Thus, their contribution to open source projects is the result of firms' deliberate decisions to finance the development of open source software. In addition there are several examples of companies that have made available formerly proprietary software as open source software. It is then unlikely that fun-seeking, altruism, quest for status and other such "soft" motives alone explain participations in open source projects.[12]

Perhaps the most important economics-based explanation to the open source paradox is based on delayed benefits earned by the open source programmers. In Lerner and Tirole (2004), developers of open source programs acquire a reputation, which is eventually rewarded in the job market. Reputation and signaling can arise as open source projects are organized such that every significant contribution can be traced back to the original author. For instance, in one of the largest projects, the Linux kernel, there exists a public "changelog" file which lists all those programmers who have contributed to the official source and their specific inputs. Each proposal to modify the code undergoes a peer review process and only those modifications sanctioned by the referees make their creators legitimate authors. The authors' names and contributions are recorded in the changelog file which is an honoring and a sign of expertise among the programmers.[13] A spot in the credits thus serves as a valuable signal on job and capital markets characterized by asymmetric information. It then pays off to extend effort on OSS programming. The ex ante expected value of the deferred payoff makes striving for the signal worthwhile since the unrestricted access to the Linux kernel code and its changelog file allows for the right interpretation and honoring even by outsiders ex post. Contributions to open source are not only unselfish donations or the pursuit of ego gratification, but also investments based, e.g., on future career concerns.

More specifically, Lerner and Tirole (2004) assume that a programmer participates in a project, whether commercial or open source, only if she derives a net benefit (broadly defined) from engaging in the activity. The net benefit is equal to the immediate payoff (current benefits minus current costs) plus the delayed payoff (delayed benefits minus delayed costs). Lerner and Tirole show that the participation question

---

[12] So far, much less attention has been devoted to firms' open source activity than that of individuals.

[13] For more on this, see Weber (2004), especially Chapter 4.

can be explained with standard economic arguments. Lerner and Tirole focus on two aspects of open source development: the immediate benefits obtained by open source developers in “scratching an itch”, namely solving a problem they face, and the existence of a “signaling incentive” that derives from the gratification associated to peer recognition or from delayed benefits in the form of better job offers relative to non-open source developers.

Once both the immediate and delayed benefits are included in an individual’s cost and benefit calculation, voluntary contributions are not paradoxical. Open source is a better mean for acquiring reputation than proprietary mode of programming. The costs incurred to build up a reputation, including both the effort spent and the opportunity costs of forgone compensation of not focusing on one’s primary mission, are lowered by the private benefits of getting a bug fixed, the “alumni effect” – programs available for free are often used as teaching tools at universities – and the enjoyment inherent in a challenging and/or just fun activity. The openness of the source code allows better performance measurement and increases the visibility of one’s contribution to the peers and to others. Open source then fulfills the main requirements spelled out in the literature on signaling incentives (e.g., Holmström, 1999).

However, there are some problems with the story based on signaling and reputation. First, note that the model does not explain why a programmer should initiate an open source project in the first place, and voluntarily publish code that might have significant commercial value. Lerner and Tirole briefly address this issue, but focus on the case in which the programmer is employed by a corporation and is prevented from taking the code private. The models by Mustonen (2003), Bessen (2002), Johnson (2002) and Kuan (2002), briefly discussed below, provide partial remedies for this problem. Second, McGowan (2002) and Weber (2004, Chapter 5) note that we should observe significantly more direct challenges to project leaders’ authority and more “strategic forking”, should reputation be the primary incentive.[14] Reputation and status are significant only in community of peers, which implies that net consumption of status is basically zero, so that increasing the status of one player reduces that of some others. However, forking is a rare event and there is strong social pressure against forking. Third, many open source projects have hierarchical structures. The possibilities a programmer is able gain status and reputation depends cru-

---

[14] Forking occurs when an individual takes the core code, incorporates his/her modules in it, sets the package up as a “new” open source project, and invites others to and join the new project. A strategic fork occurs not for technical reasons, but to create a new project that the forker could lead.

cially on the project maintainer's discretional decision (McGowan, 2002). For instance, Linus Torvalds is the "captain" of Linux kernel project and has "lieutenants" whose decisions influence an individual's opportunities to signal and seek reputation (see Weber, 2004, Chapter 4).

Empirical analyses of the open source community seem to have mixed evidence on signaling and reputation as motivators. Osterloh et al. (2003) argue that if developers really competed for reputation, we would expect them to try to heighten their visibility by submitting numerous contributions. Kogut and Meitu (2001) analyzed the "Changes" files to the Apache between March 1995 and February 2000 that lists the new patches included in each new version of Apache with their authors, and found that 82 % of individuals made only one or two contributions, which seems to contradict the reputation and signaling hypothesis. Dempsey et al. (1999) found that a majority of participants of open source projects does not contribute any code, but rather comments and explains applications. This behavior does not seem to be consistent with signaling and reputation hypothesis.

Hann et al. (2002, 2004) analyzed panel data covering a four-year period of contributors to the Apache http Server Project. They find mixed evidence for extrinsic motivations. Increases in human capital, measured by project contributions, do not lead to increased wages. In contrast, credentials earned through a merit-based ranking system are associated with significantly increased wages. They interpret this result as signaling high productive capabilities of programmers, consistent with modern labor economics.

Lakhani and von Hippel (2003) analyzed the online help system for Apache, and found a large degree of voluntary effective online support. This behavior does not seem to be based on external benefits such as reputation. Last, the surveys of open source communities (Gosh et al., 2002; Hars and Ou, 2001; Hertel et al. 2003; Lakhani and Wolf, 2001) suggest that reputation is not the most important motive for contributing to open source projects.[15]

There are also some other economics-based stories that try to explain voluntary contributions of open source software. Mustonen (2003) offers an explanation based on increased compatibility between proprietary and free programs that increases the value of the proprietary program, providing incentives to support open source efforts.

---

[15] Obviously, these types of survey results reflect programmers' self-assessment bias.

A motive to provide open source can arise from "scratching an itch" – solving a software problems such as bugs. Bessen (2002), Johnson (2002) and Kuan (2002) note that individual programmers know their own preferences and needs better than a firm does. User needs can then provide incentives to create the software in the first place, and as the costs publishing the code are very low, the decision to distribute the code may not be paradoxical after all. Bessen's (2002) explanation hinges on the complexity of the software. In Johnson (2002), a greater skill set belonging to the community of programmers as a whole can be exploited. Kuan (2002) models open source as customer vertical integration into production.[16] There have already been explanations in other fields based on horizontal user innovation networks, i.e., innovation by lead users and revelation of innovations to manufacturers or other users, (Allen, 1983; Morrison et al, 2000; von Hippel, 1988, 2002). Costs of releasing software are low due to the Internet, and the opportunity costs might be rather low for at least to students. There are two types of benefits gained from releasing software: network effects from the expansion of the user base and from the number of programmers that can report bugs, making additions to the source code, etc. (Lerner and Tirole, 2002; von Krogh, 1998, 2002). Then, the fact that user innovation networks arise is not that paradoxical.

Heterogeneity of user needs provides an important twist that actually strengthens the potentially weak reputation-based incentives. A user/programmer has basically three options, he/she can either buy a proprietary product that solves her needs, do the programming herself or wait for somebody else to develop an appropriate piece of software and to publish it on the Internet. Note that until enough modules have been published, a large-scale project, such as Linux, is essentially useless, and all users must bear the opportunity costs of waiting for the completed-enough package. Open source projects are then waiting games. Heterogeneity of user needs means that the probability of being able to free ride on the efforts of others is low enough so that a programmer may find it worthwhile to develop a required module by herself. Stenborg (2004) shows that the equilibrium of such waiting games can result in very rapid development of open source code without much waiting.

In sum, the current empirical evidence confirms that the pursuit of rewards does play a role in motivating open source developers. However, the stories based on reputation, signaling and other such indirect rewards may not present the full picture. More empirical work is needed to provide a better understanding of this paradoxical phe-

---

[16] See also Lee et. al. (2003), Leppämäki and Mustonen (2003), and Weber (2004).

nomenon. Further, there is not much empirical work targeted at the motivations of firms involved in open source.

3 Coordinating Open Source Software Projects

The second issue in open source that has attracted attention is the coordination of open source development. Software can be and open source software typically is modular. A single program, such as Linux, is assembled by combining small and independent components that communicate through interfaces. Modularity ensures that any change or addition to the source code of a module will affect the system in very limited fashion, if at all. This allows the developers to work on different components at the same time without fear of conflicts or interference. This parallel development is claimed to an increase the programming speed and the quality of final programs (see, e.g., Feller and Fitzgerald, 2002). However, there is also the potential for wasteful duplication of effort, as only one of the solutions proposed will be incorporated into the official release of a given open source product. The problem can be mitigated by the "release early, release often" principle, which leads to better information about the code under development.

Eric Raymond (1998a) has characterized the open source projects as "bazaars" in contrast to the "cathedral model" of commercial development. By bazaar, Raymond means a system of distributed innovation with a large numbers of developers and without ex ante fixed plan for development. Further, in a bazaar, decision making is distributed and especially the direction of the development is not centrally guided; the users are integrated into the production. The programmers select themselves the tasks they want to pursue.

The bazaar is not an anarchy but a sort of voluntary hierarchy. The common forms of coordination in open source projects are "benevolent dictatorship", "rotating dictatorship" and "voting committee" system (Raymond, 1998b; Ljungberg, 2000). The Linux project is an example of the fist category. Linus Torvalds is the "captain" of the project, in that he is entitled to the last word on the final design of the program, with the help of "lieutenants", each responsible for a subsystem and its interface with the rest of the project. The status of project co-developer is merit-based with the idea that "authority follows responsibility" (Raymond, 1998b). The development of Perl, a programming language, has the rotating dictatorship. Voting committee of co-developers is the coordination model adopted by a many large project, e.g., Apache http server, in which decisions are taken on the basis of a system of e-mail voting based on minimal quorum consensus (Fielding, 1999). Conflicts tend to be solved according to “authority follows responsibility” and “seniority wins”. Rule enforcement depends on the collaboration of the community, in practice either by flaming – blaming and mocking of individuals for their behavior – and shunning – refusing collaboration (Raymond, 1998b; Weber, 2004). The enforcement of the norms is also a public good. While monitoring is easy, actual punishment can only rely on informal sanctions (Kollock and Smith, 1996).

The synchronized process of design and debugging and the integration of users into development are claimed to be the two sources of efficiency gains relative to proprietary style of software development (see, e.g., Harhoff et al., 2000; Johnson, 2002; Kogut and Methiu, 2001; and Kuan, 2002). While the outcome of the innovation process is a public good, the process itself possesses benefits that can be privately appropriable. Then, free-riders are not able to obtain exactly the same benefits as those who have contributed to the innovation (von Hippel and von Krogh, 2003). Then there is then only a limited need for monitoring and high-powered incentives.

Empirical analyses seem to contradict the bazaar idea in at least two ways. First, the median number of developers in a project is very low, four in Krishnamurthy (2002) and one in Healy and Schussman (2003). In addition, Krishnamurthy (2002) finds that the level of contact and communication within open source communities is rather low. Second, open source software development does not seem to be characterized by “flat network of interacting peers” where each contributes to multiple projects, as typically only few developers actually contribute. For instance, some surveys have found that only 10 % of the total number of contributors accounted for 72.3% of the total code and the vast majority of contributors involved in only one or very few projects (Ghosh and Prakash, 2000; Ghosh et al, 2002). The analyses of the SourceForge[17] database conforms that the distribution of activity is highly skewed, regardless whether activity is measured by the number of developers, downloads, site views, etc. (Healy and Schussman, 2003; Hunt and Johnson 2002).

Also empirical analyses of single projects seem to confirm the results above. Mockus et al. (2002) found the existence of teams of 10 and 15 developers in the Apache and the Mozilla projects, respectively. They also report that the number of people who fix bugs is an order of magnitude larger than the size of the core team, and the number of people who report bugs are an order of magnitude larger than those who

---

[17] http://sourceforge.net/index.php

fix them. Koch and Schnider (2002) find similar results for the GNOME project – 52 developers account for 80 % of the source code. von Krogh et al (2003) study joining behavior and contributions to the Freenet community. They find that joiners tend to observe the development of the project before starting to contribute and that new-comers derive benefits from specializing, and that only the core developers who have invested in learning the software are able to perform less-specialized tasks such as the integration of modules.

This suggests that open source development can be characterized by a set of hetero-genous contributors with different roles. There are small number of members in the core group of developers, somewhat larger number of developers who contribute less regularly, and large number of users of who test the program, report bugs or submit suggestions.

There have been some more economics-based theoretical attempts to understand coordination in open source development. For instance, McGowan (2002) argues that transactions costs and asset specificity may be part of the explanation for coordina-tion of projects that attract a high number of developers. The idea is based on the theory of firms and other hierarchies, according to which employees are more willing to accept rules when their productivity depends on access to firm-specific assets.[18] Similarly, programmers will be more likely to accept hierarchy in projects that pro-vide unique or unusually valuable returns. The intuition should also apply to human capital specificity: programmers are willing to accept hierarchy as threat of being cut off from access to the other programmers' human capital (tacit skills, knowledge and experience).

Benkler (2003) develops a theory of commons-based peer production and compares it to markets and hierarchies. His focus is in the effectiveness of identifying and allo-cating human creativity and in transactions costs. According to Benkler, peer produc-tion has an advantage, relative to markets and hierarchies, in identifying the best available human capital in highly refined increments and allocating it to projects. De-clining price of physical capital involved in information production and the declining price of communications lower the cost of peer production and make human capital the primary economic good involved. These trends both lower the cost of coordina-tion and increase the importance of peer production's relative advantage.

---

[18] See, e.g., Grossman and Hart (1986), Hart and Moore (1990) and Hart (1995).

Stenborg (2004) analyzes the coordination problem by applying a model of generalized war of attraction, which is based on economics of auctions. He shows that the equilibrium of such waiting games can result in very rapid development of open source code without much waiting. The model, based on heterogeneity of user needs and talent, explains the coordination of open source without market prices and formal hierarchical control. The opportunity costs of waiting – time is money – and the private benefits from module developed and from completed software package act as coordinating devices.

4 Licensing

Licenses play a crucial role with respect many issues including motivation of developers, coordination of projects, effectiveness of open source development mode and relationships with commercial firms. It is then useful to first review the various software licenses and their histories used in open source.

4.1 Open Source Licenses

The Open Source Initiative[19] formally accepts licenses that fulfill Open Source Definition as open source licenses. The definition is essentially based on the substance of copyright law, but its basic logic turned upside down: the exclusive rights are reversed into non-exclusive. All major components of copyright – copying, distribution and modification – must explicitly be allowed in open source licenses. More specifically, the definition requires that a license should allow 1) free use, meaning that there are no discriminating restrictions on, e.g., commercial use, the number of users or hardware; 2) copying and distribution without any royalties, meaning that licensing fees are not a viable business model; 3) modification without any royalties (however, it is possible to include other conditions on modification such as the requirement to publish all modifications); and 4) open and easily available source code (but not necessarily gratis), which is a practical requirement to do any modifications. The most important licenses are the BSD and the GNU GPL licenses, discussed below.

---

[19] Open Source Initiative introduced the term open source and certifies licenses which comply with the general terms of their Open Source Definition.

BSD

In the 1980s, the ideas of liberal distribution terms and available source code were codified and became popular with two major operating system projects, BSD and GNU/Linux. The Open Source Initiative later introduced the umbrella term open source to describe different types of free licenses in the late 1990s. In academic circles software had been for a long time developed with the principles of open source code and free distribution. Many universities chose to use AT&T's UNIX operating system, which was licensed to educational institutions with full source code under a trade secret agreement. Users were encouraged to develop the system further, as AT&T did not really support the system. An evident implication of AT&T's policy was that UNIX became the basis for the first large-scale open collaboration development network.

A major variant of AT&T's UNIX was developed at Berkeley. Berkeley Software Distribution (BSD) soon became the academic UNIX development platform. Users sent their hacks, patches and fixes to Berkeley. If they were accepted, the contributed code was added to the BSD code base. To avoid the problems with possible copyright violations, license fees were paid to AT&T for any distribution of UNIX variants. An independent creation from Berkeley called Networking Release 1 saw the daylight in June 1989 and was distributed under the first modern BSD license. The success of Networking Release 1 raised the idea to start a project to release the whole operating system. An almost full version of BSD carrying name "Network release 2" . Later, BSD development split into different paths including FreeBSD, NetBSD and OpenBSD, who all run on the PC architecture (Välimäki, 2004).

GNU

Richard Stallman started his GNU project publishing GNU manifesto and founding the Free Software Foundation. The aim was to write a complete operating system. Stallman wrote Emacs General Public License in 1988. The idea of copyleft was for the first time implemented in GNU Emacs copyright license: it was free to copy and distribute but it was not allowed to change the license terms in any derivative work. With an innovative license Stallman was able to go against copyright with the help of copyright itself. In 1989 Emacs GPL license text was partly rewritten for clarity and the license was renamed to GNU General Public License. It became the default license for all GNU programs. The second version of GNU GPL was published in 1991 and the third version is currently in preparation. The breakthrough of GPL was not

Stallman's GNU software but a new operating system kernel Linux, which was released with GPL. The subsequent success of Linux accompanied with GNU and other free software meant that also GPL license became more known and popular.

GNU LGPL differs from GPL in functionality. It is a standard copyleft license but does not have the "viral effect". This means that direct modifications to LGPL software itself must be redistributed under LGPL (or GPL) but combinations of LGPL software with other software can be distributed with commercial license terms (Välimäki, 2004).

Some Comparisons

The GPL license requires that the source code is made available and that other programs that incorporate code from a GPL licensed program must also make the source code fully available under the GPL. Hence, programs that use code from a GPL licensed program cannot become proprietary software. The LGPL license is less restrictive than the GPL. Other moderately restrictive licenses include the Mozilla, MPL, and NPL licenses. All of these are copyleft-type of licenses. The main alternative is the BSD type license which has fewer restrictions than the GPL and LGPL licenses; it is often referred to as a non-copyleft license. For example, commercial products can be developed using software licensed under a BSD license as long as credit for the underlying code is given to the University of California. Other nonrestrictive licenses are MIT license, Sun Industry Standards Source License, Intel OSL and Apache Software License.

According to Välimäki (2004, p. 96), the GPL license is used with 66 % of the licenses on projects hosted at SourceForge, followed by LGPL (11 %), BSD (7 %), Public domain (3 %), and Artistic, MIT and Mozilla (less than 2 % each). He summarizes the functional features of some of the most popular licences as follows

	Free Distribution	Free use	Open code	Standard copyleft	Strong copyleft	Network copyleft
Proprietary	-	-	-	-	-	-
Shareware	X	-	-	-	-	-
Freeware	X	X	-	-	-	-
BSD, MIT, Apache	X	X	X	-	-	-
LGPL, MPL, ...	X	X	X	X	-	-
GPL, PL,...	X	X	X	X	X	-
AfferoPL, OSL, ...	X	X	X	X	X	X

and the rights granted as follows

	Attribution	Endorsement	Patent lic	Trademark	Warranty
MIT	Yes	-	-	-	No
BSD	Yes	No	-	-	No
(L)GPL	Yes	-	Implied	-	No
MPL, Apache...	Yes	No	Explicit	No	No
CC	Option/Yes	-	-	-	Yes/No

4.2 Trade-offs

The licensor, who may be a single developer, a group of developers with similar needs or a corporation, wants to start an open source project. The choice of the initial license may be one of the key decisions of the overall design. For instance, the license will influence whether the project will appeal to programmers. To understand licensing choices, we need to first understand the trade-offs between the various licenses.[20]

A license choice that is privately optimal for the licensor need not be socially optimal. The choice of a license impacts the community of programmers, as the benefits from working on the project may depend on the choice of license. Any negative externality

[20] This Section is largely based on Lerner and Tirole (2005).

cannot be too large, since the licensor must secure the participation of the community. Obviously, this does not imply that the preferences of the licensor and the community are perfectly aligned. Licensing will also have an impact on the end users, who may care about possible incompatibilities among versions or about the number of available applications, say. It also affects the likelihood of forking and the incentives of application developers. Licensing also affects other open source projects that later will compete with or complement the project. For instance, a program with a GPL-license may prove of no use for another open source project licensed under a BSD-license that could otherwise have made use of the program. Also software vendors and support providers opportunities are affected by the license.

When selecting a license, the licensor assesses the various benefits that the open source project will bring to her.[21] These include the intrinsic motivations discussed above, the signaling benefits such as ego gratification and career concerns, such as future job offers and access to venture capital, the need to solve concrete problems for one's employer, and the possibility of material benefits. For individuals, the latter includes the option of building a business operation on the open source code. This material incentive is distinct from the career concerns as it depends on the commercial rewards being associated with the initial open source project.

Many of the signaling benefits arise even if the subsequent work of the programmer is unrelated to the open source project. For corporations, material benefits include the increased profit on services or software that complements the open source software and the emancipation from the mark-ups and conditions imposed by a dominant software vendor with whom the open source project is meant to compete. This mixture of motivations implies that the licensors have a wide variety of goals. For example, material benefits are paramount when licensors are corporations. Such benefits provide a lesser motivation in the case of individual licensors. The licensor must then assess how her mixture of motivations and project characteristics – the environment, the size of the initial code base, the intended audience, etc. – affects various other parameters.

The open source project under consideration may not succeed on a stand-alone basis and may need complementary products. The choice of a license affects the ease with

[21] The choice of a license may be affected by issues outside utility-maximization, or may be distorted by a misunderstanding of the implications of the alternative licenses. An example of the former is the influence of ideological views; e.g., the belief that “software should be free” is sometimes invoked in favor of the GPL license.

which the different pieces of software can be combined. The advocates of the BSD license argue that the GPL and related licenses discourage potential commercial users. A case in point is the choice of license when trying to establish the software as a standard. Although unrestrictive licenses involve risks of hijacking (discussed below), here they make more sense than restrictive ones. This conjecture leads us to anticipate that projects geared toward the Internet, where standard setting has been particularly important in recent years due to the immaturity of key technologies, might be less likely to have highly restrictive licenses.

The licensing choices may also give rise to dynamic strategic complementarities or dynamic network externalities among open source licensors. If existing complementary projects have restrictive licenses, the licensor is more likely to choose a restrictive license in the anticipation of future user benefits from combining the end results. Conversely, a project with a restrictive license may not flourish in an environment dominated by BSD-licensed projects. The "greenfield" considerations need to be augmented by an analysis of "legacy aspects". An additional complication is introduced by the asymmetry of the licenses, especially the greater restrictions in the GPL license. If a BSD-licensed project wanted to make substantial use of a program (or portion of a program) covered by the GPL, the project leaders would need to obtain permission from the copyright owner (for instance, the Free Software Foundation). Were the leaders of the BSD-licensed program to incorporate the GPL code without permission, their BSD product would effectively be converted into a GPL product. Thus, they will be reluctant to add such features.

Advocates of restrictive licenses argue that unrestrictive licenses are prone to "hijacking" by commercial software vendors. The commercial firm may add some proprietary code to the open source software and take the whole private. While the resulting software may (or may not) be superior, the firm disrupts the dynamics of the open source project by de facto privatizing it.[22] While such hijacking need not be socially detrimental, the action deprives the open source contributors of some of the benefits from the project.[23] This prospect may discourage potential contributors in the first place. The argument for restrictive licenses is then that community members make project-specific investments, and hijacking poses the possibility of hold-up (see

---

[22] The original project will not be closed source nor privatized, but there is a risk that the proprietary derivative work will confuse, and perhaps dominate, the market.

[23] For instance, they may have to pay for the final software and be unable to tailor it for their own needs. Second, contributors enjoy dynamic network effects – see Lerner and Tirole (2002) – and these may be reduced by competition from a proprietary variant.

Williamson 1975, 1985). For instance, the members may lose the ability to shape the project to meet their particular needs, and their contributions become less visible because the open source community may lose interest in the project. Several covenants in the restrictive licenses can then be seen as a contractual response to the danger hold-up problem.

The risk of hijacking under alternative licenses depends on the nature of the project. Open source projects that are conservative reimplementations of pre-existing software are probably less subject to hijacking than innovative software products. Bezroukov (2002) puts Linux in the former category, and scripting languages (TCL, Perl, Python, PHP) in the latter. Another potential determinant is the size of the code. Large projects are more costly to rewrite, and so costs and delay factors may make the choice of license more relevant in this case.

An argument in favor of unrestrictive licenses is that permissiveness is what it takes to attract commercial software developers to write applications that enhance the value of the open source code. In particular, it has been suggested that in mature projects, when the energy of the initial contributors may be fading, the involvement of commercial contributors may be critical to success (Bezroukov, 2002). There are benefits in the form of reduced transaction cost to the licensor who adopts a familiar license rather than an innovative but unfamiliar one. Licensors choosing a well-known license economize on the learning costs incurred by the community as to how the license works and what its likely implications for the development process are.

4.3 Analyses of Licensing Choices

There already exist few analyses of the licensing choice for a business firm, much of it in law and economics literature.[24] Hawkins (2004) explores the tradeoffs between copyleft and non-copyleft licenses in a simple economic model. He asks why profit seeking behavior would result in the release of intellectual property to competitors (and the rest of the world). He shows that if there is no competitor to take advantage of the code, there is no incentive for the firm to not release the changes (as is the case with a viral license). If the only other player is the public, which can either ignore the code or contribute and release changes, the open source option is a dominant strategy. If the viral license can only be used by a subset of the public, open

---

[24] See, e.g., Perens (1999), Horne (2001), Rosen (2001b), Nadan (2002), Oksanen and Välimaki (2002), Stinebruner, Humphrey and Davis (2002), and Välimaki (2004)

source with a non-viral license remains the dominant strategy. However, with competition present, the equilibrium choice is a viral license. If the developer chooses a non-viral license, the competitor can take the software, inflict some losses with no offset to the firm. The viral license forces the competitor to release changes, which (in his case) yields the same payoff for the firm as if the competitor had ignored the software.

Bonaccorsi and Rossi (2003c), Lerner and Tirole (2005) and Fershtman and Gandal (2004) analyze the licensing choices empirically. Lerner and Tirole (2005) and Fershtman and Gandal (2004) examined the choice of licenses using a database of open source projects from the SourceForge web site. Bonaccorsi and Rossi (2003c) surveyed 146 Italian firms. They found that firms that employ restrictive licenses (i.e., copyleft-type) supply on average fewer proprietary products than firms that do not employ restrictive licenses, and firms that use restrictive licenses attach greater importance to "social motivation." They also note that firms commonly use mixed licensing. The possibility of dual licensing, i.e., releasing the software both under an open source and a proprietary software license, has been also theoretically analyzed, e.g., by Rosen (2001b) and Välimäki (2003).

Lerner and Tirole's dataset consisted of all software development projects listed on (and for a subset of the analyses, hosted on) the SourceForge.net. SourceForge contained (as of May 2002) approximately 39,000 projects. The sample was dominated by early-stage projects. Lerner and Tirole hypothesize that in settings where software has limited community appeal, the leader will need to offer a restrictive license in order to induce participation. This should occur in three instances: the community distrusts the licensor, the benefits from tailoring the code for particular applications are weak, or ego gratification and career concerns incentives do not have much power, as the audience mostly does not look at the code and is not composed of the programmers' peers. Lerner and Tirole find that restrictive licenses are more likely to be adopted when the software is directed at end-users, whereas less restrictive licenses are more frequently adopted for projects geared to developers, the Internet, or proprietary operating systems.

Fershtman and Gandal (2004) use a data set consisting of 71 open source projects hosted at the SourceForge web site. The sample was observed over an eighteen-month period from January 2002 through the middle of 2003, with data collected at two-month intervals, to yield a panel with nine observations for nearly all projects. They employ a random effects model. The 71 projects in the sample were chosen (in January 2000) from more than 31,000 projects that were listed at the SourceForge site at the time by selecting the most active projects in the top-level list of "topics" at SourceForge. Fershtman and Gandal examine how the type of license (and some other factors) affects the output per contributor in open source projects. They distinguish between three levels of license restrictiveness, Very restrictive (GPL), Moderately restrictive (LPGL and also Mozilla, MPL, and NPL licenses) and Non-restrictive (BSD and also MIT, Sun Industry Standards Source License, Intel OSL and Apache Software License).

Fershtman and Gandal's main result is that the output per contributor is much higher when licenses are less restrictive. This result is consistent with the hypothesis that status or self-esteem is a key source of motivation for participation in open source projects with restrictive licenses. The intuition is as follows. Potential contributors have strong incentives to contribute up to the minimum threshold level in order to be included in the "list of contributors." Once someone is included in the list of contributors, the incentive to contribute beyond that level is diminished considerably. This result supports an ideological or status motivation in the programmer's decision to participate in a project with a restrictive license. This is because status and signaling may be obtained simply by being on the list of contributors and less by the size or significance of the contribution.

Even if SourceForge is the largest open source development site and it hosts a very large number of open source projects, there are few problems with its use in empirical work. Some of the well known open source projects such as Linux (operating system), Apache (web server), Bind (A domain name software), Perl (a scripting language), and SendMail (an email server) are not hosted at SourceForge. It is possible that these source projects differ from the ones hosted at SourceForge in important ways. For example, of the above five programs, only Linux was exclusively released under the GPL, and the other four projects use BSD-type licenses. It would be interesting to empirically compare successful open source projects (typically not hosted at SourceForge) with projects hosted at SourceForge.

Howison and Crowston (2004) note that there is a large amount of anonymous data in the SourceForge system that cannot be attributed to any individual participant. There is older data that has been 'dumped' into the system, yielding valid yet totally inaccurate data. Also, SourceForge has become the 'repository of record' for the OS community, yet for important projects it is not the repository of use. For instance, vim, an important programmer's editor, is listed at SourceForge but has only 3 developers, 0 % activity and has not released any files. The page is simply a placeholder that points to the vim 'repository of use'.

5 Public Policies

Open source software has also attracted policymakers' attention; see, e.g., papers in Hahn (2002). One of the issues is, should governments subsidize open source. The question arises from the possibility of significant market failures on software markets. There are a number of potential sources of market failures in software market: there are large supply-side economies of scale, innovation and competition for market, not competition on a market, play a crucial roles, and there are network effects, i.e., demand-side scale economies (Bessen, 2002b; Comino and Manenti, 2003; Evans and Reddy, 2002; Lessig, 2002; Schmidt and Schnitzer, 2002; and Smith, 2002).

One should then compare the relative advantages of open source and proprietary software in mitigating these potential sources of market failures. According to Schmidt and Schnitzer (2002), open source enjoys a clear advantage relative to proprietary software only with respect to supply-side economies of scale. Open source software is priced at or close to marginal cost, so that it ensures the widest access to the developed software, and thus maximizes static efficiency. Commercial firms, first, need to recoup the fixed development costs, and second, tend to have market power, so they charge prices above marginal costs, thereby reducing access to developed software. The benefits of open source regarding innovation and network effects are less clear or non-existent. The profit motive is likely to provide stronger incentive to innovate than the mixture of incentives present in open source projects. Further, firms might have more incentives to tailor their products to the needs of the average costumer. Evans and Reddy (2002) argue that open source is predominantly imitative, not innovative in nature. Public support of open source then leads to less innovation and development. According to Bessen (2002a, b), open source corrects imperfections in the market for proprietary software, without requiring government intervention through subsidies or procurement preferences.

Bessen (2002a, b) notes another type of market failure created by the government: software patents. The strengthening and extension of intellectual property rights over the last two decades have raised concerns about hold-up for cumulative innovation (see, e.g., Bessen and Maskin, 2002). When innovation is cumulative – new products benefit from previous innovations, possibly infringing previous patents – an early patent holder has a potential claim against subsequent innovators. Anticipating the expected cost of such claims, a second innovator may choose to perform a suboptimal level of R&D or, perhaps, not to invest in the innovation at all. The concern is that stronger patent rights may increase the occurrence of hold-up, reducing R&D incentives, thus slowing the pace of innovation. Open source is a private solution that can and should be allowed to flourish without government intervention. As a remedy, Bessen recommends the removal of a market failure the government itself has created by strengthening patent protection in the software domain.

To the best of my knowledge, there are no empirical studies analyzing public policies directed at open source.

6 Conclusion

Open source software has gained the attention of several theoretical and empirical studies, not only from software engineers but from various social sciences. Analyses have looked at the nature of individual motivations to developed open source software as well as the ideological issues related to the idea that software should be free; analyzed the degrees of complexity and modularity, which affect the costs of coordination and the hierarchical organization used in open source development; looked at the licensing choices and the related public policies.

There are many directions for further research. First there are the questions regarding individual incentives. Some explanations for the public good paradox have been presented, but the economics analyses have only touched the ideas presented by other social scientist, such as status and other intrinsic motivations. Individual incentives are likely to affect the choice of projects to be developed and the outcome of the development effort. Second, economics approach has made only few attempts to look at governance of open source projects. It is not entirely clear what explains the coordination, authority and hierarchy in open source communities, which are characterized by voluntary collaboration, and where the usual elements of the governance of firms are based upon are lacking. Third, licensing choices have been analyzed almost exclusively from the developers' point of view. For instance, what are the effects of the copyleft licenses for social welfare?

Fourth, the interactions between commercial actors and open source community have not received much theoretical nor empirical economics attention. For instance, the FLOSS-project (Ghosh et al., 2002) looks at licenses of many large and popular projects, discuss some software characteristics, the software value chain, open source at the market place and business models based on open source. However, not much analysis was presented on the interactions between open source and commercial worlds. Fifth, as Ghosh et al. (2002) in their FLOSS-study note, by 2002, IBM claims to have spent $1 billion on Linux alone and is also active in several other open source projects. Other companies also devote considerable resources to the development of OS software. Despite this size, the companies' motivation behind their open source engagement is not as well understood as the motivation of individual developers. Sixth, the issue of diffusion of open source programs has not been analyzed at all. Last, there is the question, is open source particular to software only, or are there more general lessons to be learned and applied in some other fields? There are some analyses of other activities such as Wikipedia, Kuro5hin, NASA Clickworkers, Slashdot and Project Gutenberg that seem to use open source style in their development and production efforts (see, e.g., Benkler, 2003).

References

Allen, Robert C. (1983), “Collective Invention”, *Journal of Economic Behavior and Organization* 41, 1-24.

Auriol, E. and R. Renault (2003), “Status and Incentives”, mimeo, available at http://www.idei.asso.fr/Commun/Articles/Auriol/status_incentives.pdf.

Benkler, Yochai (2003), “Coase’s Penguin, or Linux and the Nature of the Firm”, *Yale Law Journal*, 112, Winter 2002–2003.

Bergquist, M. and J. Ljungberg (2001), “The Power of Gifts: Organizing Social Relationships in Open Source Communities”, *Information Systems Journal* 11, 305-20.

Bessen, James (2001), “Open Source Software: Free Provision of Complex Public Goods” mimeo, available at www.researchoninnovation.org/opensrc.pdf.

Bessen, James (2002a), “Hold-up and Patent Licensing of Cumulative Innovations with Private Information”, mimeo,

Bessen, James (2002b), “What Good is Free Software?”, in R.W. Hahn (ed.), *Government Policy toward Open Source Software*, AEI Brooking Joint Center, Washington D.C., available at http://www.aei.brookings.org/publications/abstract.php?pid=296.

Bessen, J. and E. Maskin (2000), “Sequential Innovation, Patents and Imitation”, mimeo, MIT Economics Department.

Bezroukov, Nicolaj (1999) “Open Source Software Development as a Special Type of Academic Research (Critique of Vulgar Raymondism),” *First Monday*, 4:10, available at http://www.firstmonday.org/issues/issue4_10/bezroukov/

Bonaccorsi, Andrea and Cristina Rossi (2003a), “Why Open Source Can Succeed”, *Research Policy*, 327, 1243-58.

Bonaccorsi, Andrea and Cristina Rossi (2003b), “Contributing to Common Pool Resources. A Comparison Between Individuals and Firms”, mimeo, Sant Anna School of Advanced Studies, available at http://opensource.mit.edu/papers/bnaccorsirossidevelopers.pdf.

Bonaccorsi, Andrea and Cristina Rossi (2003c), “Licensing Schemes in the Production and Distribution of Open Source Software. An Empirical Investigation”, mimeo, Sant Anna School of Advanced Studies, available at http://opensource.mit.edu/papers/bnaccorsirossilicense.pdf.

Brooks, F. (1975), *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley.

Cohendet, P.; F. Creplet, and O. Dupouët (2001), “Organisational Innovation, Communities of Practice and Epistemic Communities: The Case of Linux”, in A. Kirman and J.B. Zimmermann (eds.) *Economics with Heterogeneous Interacting Agents*, Springer, Berlin.

Deci, E.L. (1980), *The Psychology of Self-Determination*, Lexington, MA: Lexington Books.

Deci, E.L. and R.M. Ryan (1985), *Intrinsic Motivation and Self-Determination in Human Behavior*, Plenum Press, New York.

Dempsey, B.J.; J. Greenberg, P. Jones and D. Weiss (1999), “A Quantitative Profile of a Community of Open Source Linux Developers”, SILS Technical Report TR-1999-05, available at http://www.ibiblio.org/osrt/develpro.html

Di Carlo, Lisa (2002), “Linux Not Just for Geeks Anymore”, *Forbes* July 15, 2002, available at http://www.forbes.com/2002/07/15/0715linux.html

Evans, D.S. (2002), “Politics and Programming: Government Preferences for Promoting Open Source Software”, in R.W. Hahn (ed.) *Government Policy toward Open Source Software*, AEI Brooking Joint Center, Washington D.C., available at http://www.aei.brookings.org/publications/abstract.php?pid=296

Evans, D.S. and B. Reddy (2003), “Government Preferences for Promoting Open Source Software: A Solution in Search of a Problem”, *Michigan Telecommunications and Technology Law Review* 9:313, available at http://www.mttlr.org/volnine/evans.pdf

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

Fershtman, C. and N. Gandel (2004), “The Determinants of of Output per Contributor in Open Source Projects: An Empirical Examination”, CEPR Discussion Paper No. 4329.

Fershtman, C.; K. Murphy and Y. Weiss (1996), “Education, Status and Growth”, *Journal of Political Economy*, 104, 108-32.

Fershtman, C. and Y. Weiss (1993), “Social Status, Culture and Economic Performance” *Economic Journal*, 103, 946-50.

Fielding, R.T. (1999), “Shared Leadership in the Apache Project”, *Communications of the ACM* 424, 38–39.

Ghosh, Rishab Aiyer (1998) “Cooking Pot Markets: An Economic Model for the Trade in Free Goods and Services on the Internet,” *First Monday* 3:3, available at: www.firstmonday.org/issues/ issue3_3/ghosh/index.html.

Ghosh, Rishab and Vipul Ved Prakash (2000), “The Orbiten Free Software Survey”, *First Monday* 5:7, available at http://www.firstmonday.org/issues/issue5_7/ghosh/index.html; 5/10/2000

Ghosh, R.A.; R. Glott, B. Kreiger and G. Robles (2002), “The Free/Libre and Open Source Software – FLOSS Final Report”, available at http://www.infonomics.nl/FLOSS/report.

Gonsalves, Antone (2003), “Windows Gains Market Share, Despite Linux Threat”, available at http://www.techweb.com/wire/story/TWB20031008S0013.

Grossman, S.J. and O.D. Hart (1986), “The Costs ad Benefits of Ownership: A Theory of Vertical and Lateral Integration,” *Journal of Political Economy* 944, 691-719.

Hahn, R.W. (ed.) (2002), *Government Policy toward Open Source Software*, AEI Brooking Joint Center, Washington D.C., available at http://www.aei.brookings.org/publications/abstract.php?pid=296

Hann, I.-H.; J. Roberts, S. Slaughter and R. Fielding (2002), “Delayed Returns to Open Source Participation: An Empirical Analysis of the Apache HTTP Server Project”, mimeo, available at http://idei.fr/activity.php?a=2495.

Hann, I-H.; J. Roberts, S. Slaughter and R. Fielding (2004), “An Empirical Analysis of Economic Returns to Open Source Participation”, mimeo, USC, available at http://www-rcf.usc.edu/~hann/publications.html

Harhoff, D.; J. Henkel and E. von Hippel (2000), “Profiting from Voluntary Information Spillovers: How Users Benefit by Freely Revealing their Innovations”, MIT Sloan School of Management Working Paper, available at www.opensource.mit.edu/papers/evhippel-voluntaryinfospillover.pdf.

Hars, A., and S. Ou (2000), “Working for Free? Motivations of Participating in Open Source Projects”, in *Proceedings of the 34th Hawaii International Conference on System Sciences*, IEEE, available at http://oss.usc.edu/research/ossabstract.pdf

Hart, Oliver D. (1995), *Firms, Contracts and Financial Structure*, Clarendon Press, Oxford.

Hart, Oliver and John Moore (1990), “Property Rights and the Nature of the Firm,” *Journal of Political Economy* 986, 1119-58.

Hawkins, R.E. (2004) “The Economics of the Open Source Software for a Competitive Firm”, *Netnomics*, forthcoming.

Healy K. and A. Schussman (2003), “The Ecology of Open Source Software Development”, mimeo, University of Arizona, available at http://opensource.mit.edu/papers/healyschussman.pdf.

Hertel G.; S. Niedner and S. Hermann (2003), “Motivation of Software Developers in the Open Source Software Projects: An Internet-based Survey of Contributors to the Linux Kernel”, *Research Policy*, 327, 1159-77.

Holmstrom, B. (1999), “Managerial Incentive Problems: A Dynamic Perspective”, *Review of Economic Studies*, 66, 169-82.

Hunt, F. and P. Johnson (2002), “On the Pareto Distribution of Sourceforge Projects”, in *Proceedings of the Open Source Software Development Workshop*, 122-29, Newcastle, UK, available at http://www.ifm.eng.cam.ac.uk/people/fhh10/Sourceforge/Sourceforge%20paper.pdf

Johnson, Justin Pappas (2002), “Open Source Software: Private Provision of a Public Good”, *Journal of Economics & Management Strategy*, 11, 637-662, available at http://opensource.mit.edu/papers/johnsonopensource.pdf.

Kennedy, D.M. (2001), “A Primer on Open Source Licensing Legal Issues: Copyright, Copyleft and Copyfuture”, *Saint Louis University Public Law Review*, XX, available at http://www.denniskennedy.com/opensourcedmk.pdf.

Koch, S. and G. Schneider (2002), “Effort, Co-operation and Co-ordination in an Open Source Software Project: GNOME”. *Information Systems Journal* 12, 27–42.

Kogut, Bruce and Anca Metiu (2001), “Open-Source Software Development and Distributed Innovation,” *Oxford Review of Economic Policy*, 172, 248-64.

Kollock, Peter (1999), “The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace”, in Marc Smith and Peter Kollock (eds.), *Communities in Cyberspace*. Routledge, London.

Kollock, Peter and Marc Smith (1996), “Managing the Virtual Commons: Cooperation and Conflict in Computer Communities”, in Susan Herring (ed.) *Computer-Mediated Communication: Linguistic, Social and Cross-Cultural Perspectives*, John Benjamins, Amsterdam.

Krishnamurthy, Sandeep (2002), “Cave or Community? An Empirical Examination of 100 Mature Open Source Projects”, *First Monday* 7:6, available at: http://firstmonday.org/issues/issue7_6/krishnamurthy/index.html.

Kuan, Jennifer (2001), “Open Source Software as Consumer Integration into Production”, mimeo, available at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=259648.

Lakhani, K.; B. Wolf, J. Bates, C. DiBona (2002), *The Boston Consulting Group Hacker Survey*, Release 0.73, available at http://www.bcg.com/opensource/BCGHackerSurveyOSCON24July02v073.pdf.

Lakhani, Karim and Eric von Hippel (2003), "How Open Source Software Works: 'Free' User-to-User Assistance," *Research Policy*, 326, 923-43.

Lakhani, Karim and Robert G. Wolf (2003), "Why Hackers Do What They Do: Understanding Motivation Efforts in Free/OSS Projects", MIT Sloan School of Management Working Paper No 4425-03.

Lee, J. (2003), "Open Source Software Licenses and Innovation", mimeo, available at: http://VirtualGoods.tu-ilmenau.de/2003/licenses_innovation_leejz.pdf.

Lee, S.H. (1999), "Open Source Software Licensing", mimeo, available at http://eon.law.harvard.edu/openlaw/gpl.pdf.

Lee, S.; N. Moisa and M. Weiss (2003), "Open Source as a Signaling Device – An Economic Analysis", mimeo, available at: http://opensource.mit.edu/papers/leemoisaweiss.pdf.

Leppämäki, M. and M. Mustonen (2003), "Spence Revisited - Signalling with Externality: The Case of Open Source Programming", mimeo, University of Helsinki.

Lerner, J. and J. Tirole (2002), "Some Simple Economics of Open Source", *Journal of Industrial Economics* 52, 197–234, available at http://www.people.hbs.edu/jlerner/simple.pdf.

Lerner, J. and J. Tirole (2005), "The Scope of Open Source Licensing", *Journal of Law, Economics, and Organization*, forthcoming, available at http://www.people.hbs.edu/jlerner/OSLicense.pdf

Lessig, Lawrence (2002), "Open Source Baselines: Compared to What?", in R.W. R.W. Hahn (ed.), *Government Policy toward Open Source Software*, AEI Brooking Joint Center, Washington D.C., available at http://www.aei.brookings.org/publications/abstract.php?pid=296.

Lindenberg, S. (2001), "Intrinsic Motivation in a New Light", *Kyklos* 542/3, 317-43.

Ljungberg, J. (2000), "Open Source Movements as a Model for Organizing", *European Journal of Information Systems*, 94, pp. 208-16.

Manenti, Fabio and Stefano Comino (2003), "Open Source vs Closed Source Software: Public Policies for the Software Market", working paper, University of Padova, available at http://opensource.mit.edu/papers/cominomanenti.pdf.

McGowan, D. (2001), "Legal Implications of Open-Source Software", *University of Illinois Law Review* 1, 241–304, available at http://ssrn.com/abstract=243237.

McJohn, Stephen M. (2000), "The Paradoxes of Free Software", *George Mason Law Review*, Fall 2000, 25-68.

Mocus A.; R. Fielding and J. Herbsleb (2000), "A Case Study of Open Source Software Development: The Apache Server", *Proceedings of the 2000 International Conference on Software Engineering ICSE2000*, Limerick, Ireland, June 4-11, 263-272, available at http://opensource.mit.edu/papers/mockusapache.pdf.

Moglen E. (1999), "Anarchism Triumphant: Free Software and the Death of Copyright", *First Monday* 4:8, available at http://www.firstmonday.org/issues/issue4_8/moglen/index.html.

Moody, Glynn (2001) *Rebel Code: The Inside Story of Linux and the Open Source Revolution*, Cambridge, Massachusetts: Perseus Publishing.

Moon, J.Y. and L. Sproull (2000) “Essence of Distributed Work: The Case of the Linux Kernel”, *First Monday* 5:11, available at http://www.firstmonday.org/issues/issue5_11/moon/index.html

Morrison, P.; J. Roberts and E. von Hippel (2000), “Determinants of User Innovation and Innovation Sharing in a Local Market ,” *Management Science* 4612, 1513-27.

Mustonen, Mikko (2003) “Copyleft – The Economics of Linux and Other Open Source Software”, *Information Economics and Policy*, 15, 99-121.

Pal, Nilendu and T.R Madanmohan (2001), “Competing on Open Source: Strategies and Practise”, mimeo, available at http://opensource.mit.edu/papers/madanmohan.pdf.

Oksanen, V. and M. Välimäki (2002), “Open Source Licensing Models for a Company Developing Mass Market Software”, *Proceeding of LawTech 2002*, Cambridge MA, USA, available at http://www.hiit.fi/de/valimaki_oksanen_lawtech_2002.pdf.

Osterloh M.; S. Rota and M. von Wartburg (2002) “Open Source – New Rules in Software Development?”, mimeo, University of Zurich.

Ostrom, E. (1990), *Governing the Commons: The Evolution of Institutions for Collective Action*, Cambridge University Press, New York, NY.

Parker, Robert and Bruce Grimm (2000), “Recognition of Business and Government Expenditures for Software as Investment: Methodology and Quantitative Impacts, 1959-98”, Paper presented at the May 5th, 2000 meeting of the BEA Advisory Committee, available at <www.bea.doc.gov/bea/papers.htm>.

Ravicher, Daniel B. (2000), “Facilitating Collaborative Software Development: The Enforceability of Mass-Market Public Software Licenses”, *Virginia Journal of Law and Technology* 115, 1522-687.

Raymond, Eric S. (1998a), “The Cathedral and the Bazaar”, *First Monday*, 3:3, available at www.firstmonday.org/issues/issue3_3/raymond/index.html.

Raymond, Eric S. (1998b), “Homesteading the Noosphere”, *First Monday*, 3:10, available at www.firstmonday.org/issues/issue3_10/raymond/index.html.

Raymond, Eric S. (1999), “A Brief History of Hackerdom”, in C. DiBona, S. Ockman, and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

Rosen, L. (2001a), “The Unreasonable Fear of Infection”, available at http://www.rosenlaw.com/html/GPL.PDF.

Rosen, L. (2001b), “Which Open Source License Should I Use For My Software?”, available at http://www.rosenlaw.com/html/GL5.pdf.

Schmidt, K. and M. Schnitzer (2003), “Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market”, CEPR Discussion Paper, No. 3793, available at http://opensource.mit.edu/papers/schmidtschnitzer.pdf

Schweik, C.M. and A. Semenov (2003), “The Institutional Design of Open Source Programming: Implications for Addressing Complex Public Policy and Management Problems”, *First Monday* 8:1, available at http://firstmonday.org/issues//issue8_1/schweik/index.html.

Shah, S. (2002), “Open Source Software as Lead User’s Make or Buy Decision: A Study of Open and Closed Source Quality”, *TIIP Newslette* 2002/3, available at http://www.researchoninnovation.org/tiip/archive/kuan.htm.

Smith, B.L. (2002) "The Future of Software: Enabling the Marketplace to Decide", in R.W. Hahn (ed.), *Government Policy toward Open Source Software*, AEI Brooking Joint Center, Washington D.C., available at http://www.aei.brookings.org/publications/abstract.php?pid=296.

Stallman, R. (1984), "What is Copyleft?", available at http://www.gnu.org/copyleft/copyleft.html.

Stallman, R. (1999), "The GNU Operating System and the Free Software Movement", in C. DiBona, S. Ockman, and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

Stenborg, M. (2004), "Waiting for F/OSS: Coordinating the Production of Free/Open Source Software", ETLA Discussion Paper.

Stewart, Katerine J. and Sanjay Gosain (2003), "Impact of Ideology, Trust and Communication on Effectiveness in Open Source Software Development Teams", mimeo, available at http://opensource.mit.edu/papers/stewartgosain.pdf.

Torvalds, L. (1998), "What Motivates Free Developers?", interview in *First Monday* 3:3, available at http://firstmonday.org/issues/issue3_3/torvalds/index.html.

Torvalds, L. (1999), "The Linux Edge", Raymond, Eric S. (1999), "A Brief History of Hackerdom", in C. DiBona, S. Ockman, and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

Valloppillil, V. (1998), "Halloween Document I (Version 1.14)" or "Open Source Software: A (New?) Development Methodology?", Microsoft internal memo, available at http://www.opensource.org/halloween/halloween1.html.

Valloppillil, V. and J. Cohen (1998), "Halloween Document II (Version 1.4)" or "Linux OS Competitive Analysis: The Next Java VM?", Microsoft internal memo, available at http://www.opensource.org/halloween/halloween2.php

von Hippel, Eric (1988), *The Sources of Innovation*, New York: Oxford University Press.

von Hippel, E. and E. von Krogh (2003), "The Private-Collective Innovation Model in Open Source Software Development: Issues for Organization Science", *Organization Science*, forthcoming.

von Krogh, G. (1998), "Care in Knowledge Creation", *California Management Review*, 403, 133-53.

von Krogh, G. (2002), "The Communal Resource and Information Systems", *Journal of Strategic Information Systems*, 112, 85-107.

von Krogh, G.; S. Spaeth and K.R. Lakhani (2003), "Community, Joining, and Specialization in Open Source Software Innovation: A Case Study", *Research Policy* 327, 1217-47.

Välimäki, M. (2004), *The Challenge Of Open Source Licensing. How Open Source Changed Thinking About Intellectual Property In The Software Industry?*, draft of an unpublished PhD thesis, available at http://www.valimaki.org/.

Weber, Steven (2004), *The Success of Open Source*, Harvard University Press.

Wheeler, David A. (2004), "Why Open Source Software / Free Software (OSS/FS)? Look at the Numbers!", available at www.dwheeler.com/oss_fs_why.html

Wichmann, T. (2002a), "Firms' Open Source Activities: Motivations and Policy Implications", *Free/Libre and Open Source Software: Survey and Study*, FLOSS Final Report Part 2, International Institute of Infonomics, Berlecom Research GmbH, available at http://www.infonomics.nl/FLOSS/report/.

Wichmann, T. (2002b), "Basics of Open Source Software Markets and Business Models", *Free/Libre and Open Source Software: Survey and Study*, FLOSS Final Report Part 3, International Institute of Infonomics, Berlecom Research GmbH, available at http://www.infonomics.nl/FLOSS/report/.

Williamson, Oliver (1975), *Markets and Hierarchies: Analysis and Antitrust Implications*, The Free Press.

Williamson, Oliver (1985), *The Economic Institutions of Capitalism*, The Free Press.

Some Other Relevant Literature

Babcock C. (2001), "Open Source Code: A Corporate Building Block", *Interactive Week*, available at http://zdnet.com/intweek/stories/news/0,4164,2717905,00.html.

Baldwin, Carliss Y. and Kim B. Clark (2003), "Does Code Architecture Mitigate Free-Riding in the Open Source Development Model?", mimeo, Harvard Business School, available at http://opensource.mit.edu/papers/baldwin.pdf.

Behlendorf, Brian (1999), "Open Source as a Business Strategy", in C. DiBona, S. Ockman and M. Stone (eds.) *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

Bitzer, Jürgen and Philipp J.H. Schröder (2003), "Bug-Fixing and Code-Writing: The Private provision of Open Source Software", mimeo, DIW Berlin – German Institute for Economic Research, paper presented at EARIE 2003.

Bobko, P.K. (2001), "Open Source Software and the Demise of Copyright", *Rutgers Computer and Technology Law Journal*, 27, 51-92.

Dahlander, Linus (2004), "Appropriating the Commons: Firms in Open Source Software", mimeo, Chalmers University of Technology, available at http://opensource.mit.edu/papers/dahlander2.pdf.

Dalle, Jean-Michel and Nicolas Jullien (2000), "NT vs. Linux, Or Some Explorations into the Economics of Free Software," in G. Ballot and G. Weisbuch (eds.), *Application of Simulation to Social Sciences*, Hermès, Paris, France.

Dalle, Jean-Michel and Nicolas Jullien (2003) "'Libre' Software: Turning Fads into Institutions?", *Research Policy*, 321, 1-11.

Dalle, J.-M.; P.A. David, R.A. Ghosh and F.A. Wolak (2004), "Free & Open Source Software Developers and 'the Economy of Regard': Participation and Code-Signing in the Modules of the Linux Kernel", mimeo, SIEPR, Stanford, available at siepr.stanford.edu/programs/OpenSoftware_David/ Economy-of-Regard_8+_OWLS.pdf

Edwards, Kasper (2001), "Epistemic Ccommunities, Situated Learning and Open Source Software Development", mimeo, available at http://opensource.mit.edu/papers/kasperedwards-ec.pdf.

Franck, E. and C. Jungwirth (2001), "Reconciling Investors and Donators – The Governance Structure of Open Source", University of Zurich Working Paper, available at http://www.isu.unizh.ch/fuehrung/Dokumente/WorkingPaper/8abstract.pdf

Franke, Nikolaus and Eric von Hippel (2003), "Satisfying Heterogeneous User Needs via Innovation Toolkits: The Case of Apache Security Software," *Research Policy* 327, 1199-215.

Gomulkiewicz, R.W. (1998), “The License is the Product: Comments on the Promise of Article 2B for Software and Information Licensing”, *Berkeley Technology Law Journal On Line* 133, available at: http://www.law.berkeley.edu/journals/btlj/articles/vol13/Gomulkiewicz/html/reader.html.

Gomulkiewicz, R.W. (1999), “How Copyleft Uses License Rights to Succeed in the Open Source Software Revolution and the Implications for Article 2B”, *Houston Law Review* 36, 179–94.

Gruber, Marc and Joachim Henkel (2004), “New Ventures Based on Open Innovation – An Empirical Analysis of Start-Up Firms in Embedded Linux”, available at http://opensource.mit.edu/papers/gruberhenkel.pdf.

Hecker, F. (1999) “Setting Up Shop: The Business of Open-Source Software”, *IEEE Software*, 16 1, 45–51.

Heller, Michael A. and Rebecca S. Eisenberg (1998), “Can Patents Deter Innovation? The Anticommons in Biomedical Research”, *Science* 280, 698-701.

Henkel, Joachim (2002), “Open Source Software from Commercial Firms: Tools, Complements, and Collective Invention”, GEABA Discussion Paper 02-27, available at http://www.inno-tec.bwl.uni-muenchen.de/forschung/henkel/OSS_JHenkel_2003-05.pdf

Henkel, Joachim and Mark Tins (2004), “Munich/MIT Survey: Development of Embedded Linux”, mimeo, University of Munich, available at http://www.inno-tec.bwl.uni-muenchen.de/forschung/henkel/MunichMIT-Survey_Embedded_Linux_2004.pdf

Horne, N. (2001), “Open Source Software Software Licensing: Using Copyright Law to Encourage Free Use”, *Georgia State University Law Review*, 17, 863-, available at http://gsulaw2.gsu.edu/lawreview/archives/fulltext.php

Järvinen, H. (2002), “Legal Aspects of Open Source Licensing”, mimeo, available at http://www.cs.helsinki.fi/u/campa/teaching/oss/papers/jarvinen.pdf.

Jørgensen, N. (2001), “Putting It All in the Trunk: Incremental Software Development in the FreeBSD Open Source Project”, *Information Systems Journal*, 11, 321-36.

Kahin, Brian (2002), “The Patent Theat to Open Source”, in *Public Interest and Policy Issues*, Report of the Georgetown University 2002 Open Source Summit, available at http://opensource.georgetown.edu/report/osreport.pdf.

Kelty, Christopher M. (2001), “Free Software/Free Science”, *First Monday* 6:12, available at www.firstmonday.org/issues/issue6_12/kelty/index.html.

Kuwabara, Ko (2000) “Linux: A Bazaar at the Edge of Chaos,” *First Monday*, 5:3, available at www.firstmonday.org/issues/issue5_3/kuwabara/index.html.

Madey, G.; F. Freeh and R. Tynan (2004), “Modeling the Free/Open Source Software Community: A Quantitative Investigation”, in Stephan Koch (ed.) *Free/Open Source Software Development*, Idea Publishing, forthcoming, available at www.nd.edu/~oss/Papers/BookChapter.pd

Mateos-Garcia, Juan and W. Edward Steinmueller (2003), ”The Open Source Way of Working: A New Paradigm for the Division of Labour in Software Development?”, Working Paper, SPRU-University of Sussex, available at http://siepr.stanford.edu/programs/OpenSoftware_David/oswp1.pdf

Maurer, Stephen M. (2003), “New Institutions for Doing Science: From Databases to Open Source Biology,” paper presented at the European Policy for Intellectual Property Conference, Maastricht.

Mazzoleni, R. and Richard R. Nelson (1998), "The Benefits and Costs of Strong Patent Protection: A Contribution to the Current Debate", *Research Policy* 27, 273-84

Meurer, Michael J. (2003), "Controlling Opportunistic and Anticompetitive Intellectual Property Litigation", *Boston College Law Review*, forthcoming.

Nadan, Christian H. (2002), "Open Source Licensing: Virus or Virtue?", *Texas Intellectual Property Journal* 10, 349-.

Neumann, P.G. (2001), "Robust Non-Proprietary Software", IEEE Symposium on Security and Privacy, May 2001.

O'Mahony, S. (2002), *Community Managed Software Projects: The Emergence of a New Commercial Actor*, unpublished Ph.D.Dissertation, Stanford University.

O'Mahony, S. (2003), "Guarding the Commons: How Do Community Managed Software Projects Protect Their Work", *Research Policy*, 327, 1179-98.

Osterhout, J. (1999), "Free Software Needs Profit", *Communications of the ACM* 424, 38–9.

Pagano, U. (1999) "Is Power an Economic Good? Notes on Social Scarcity and the Economics of Positional Goods". In S. Bowles, M. Franzini and U. Pagano (eds.), *The Politics and the Economics of Power*, Routledge, London.

Pagano, U. (2004), "Legal Positions and Institutional Complementarities", mimeo, Università di Siena, available at http://www.unisi.it/lawandeconomics/stile2004/pagano.pdf.

Perens, Bruce (1999), "The Open Source Definition", in C. DiBona, S. Ockman, and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

Rosenberg, D.K. (1998), "Evaluation of Public Software Licenses", available at http://www.stormian.com/Public Licenses.html.

Stinebruner, S.A.; T.W. Humphrey and J.P. Davis (2002), "Benefits, Risks and Considerations in Using Open Licensed Software", available at http://www.whepatent.com/pub/opensource.pdf.

West, Joel (2003), "How Open is Open Enough? Melding Proprietary and Open Source Platform Strategies", *Research Policy* 32: 1259-85.

West, Joel and Jason Dedrick (2001), "Open Source Standardization: The Rise of Linux in the Network Era", *Knowledge, Technology, & Policy*, 14(2): 88-112.

Xu, Xiaopeng (2002) "Development Costs and Open Source Software", Working Paper, University of California at Berkeley, available at opensource.mit.edu/papers/xxu.pdf.

Xu, Jin and Gregory Madey "Exploration of the Open Source Software Community", NAACSOS Conference 2004, Pittsburgh, PA, June 2004, available at http://www.nd.edu/~oss/Papers/naacsos04Xu.pdf

Young, R. (1999), "Giving it Away: How Red Hat Software Stumbled Across a New Economic Model and Helped Improve an Industry", in C. DiBona, S. Ockman, and M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, CA.

E L I N K E I N O E L Ä M Ä N T U T K I M U S L A I T O S (ETLA)
THE RESEARCH INSTITUTE OF THE FINNISH ECONOMY
LÖNNROTINKATU 4 B, FIN-00120 HELSINKI

Puh./Tel. (09) 609 900 Telefax (09) 601753
Int. 358-9-609 900 Int. 358-9-601 753
http://www.etla.fi

KESKUSTELUAIHEITA - DISCUSSION PAPERS ISSN 0781-6847

Julkaisut ovat saatavissa elektronisessa muodossa internet-osoitteessa:
http://www.etla.fi/finnish/research/publications/searchengine

No 917 MAARIT LINDSTRÖM, Ulkomaiset yritykset Suomessa: Investointimotiivit ja näkemykset toimintaympäristöstä. 16.06.2004. 38 s.

No 918 LAURA PAIJA, Allocation of Control Rights to Customised Products: Empirical Analysis of Finnish SMEs. 10.08.2004. 38 p.

No 919 MIKA MALIRANTA – PETRI ROUVINEN, Informational Mobility and Productivity – Finnish Evidence. (Tiedollisen liikkuvuuden tuottavuusvaikutukset Suomessa). 22.07.2004. 14 p.

No 920 MIKA MALIRANTA – SATU NURMI, Analyzing entrepreneurship with the Finnish linked employer-employee data (FLEED). Matching and qualitative properties of the data. 29.07.2004. 20 p.

No 921 EDVARD JOHANSSON – HANNU AHO – URPO KIISKINEN – KARI POIKOLAINEN, The association of alcohol dependency with employment probability: Evidence from the population survey “Health 2000 in Finland”. Alkoholiriippuvuus ja todennäköisyys olla kokopäivätyössä: Tuloksia “Terveys 2000” aineistosta. 02.08.2004. 28 p.

No 922 ANNIKA EVÄLÄ, Koulutuksen panokset ja tuotokset – Suomi kansainvälisessä vertailussa. 24.06.2004. 93 s.

No 923 ARI HYYTINEN – MIKA PAJARINEN, Opacity of Young Firms: Faith or Fact? 06.08.2004. 24 p.

No 924 PAAVO SUNI, Kiina maailmantaloudessa – Globaalitalouden kasvu ja teollisen työnjaon muutos, Taustaraportti (China in The World Economy – Global Economic Growth and the Changing International Division of Labour, A Background Report). 23.08.2004. 24 s.

No 925 VILLE KAITILA, The Factor Intensity of Accession and EU15 Countries’ Comparative Advantage in the Internal Market. 25.08.2004. 23 p.

No 926 ANTTI-JUSSI TAHVANAINEN – RAINE HERMANS, Financial Pecking Order and the Value Platform of Intellectual Capital. Observing the Finnish Biotechnology Industry. 25.08.2004. 34 p.

No 927 JYRKI ALI-YRKKÖ – MAARIT LINDSTRÖM – MIKA PAJARINEN – PEKKA YLÄ-ANTTILA, Suomen asema globaalissa kilpailussa – yritysten sijaintipäätöksiin vaikuttavat tekijät. 30.08.2004. 83 s.

No 928 PETRI BÖCKERMAN – EDVARD JOHANSSON – SATU HELAKORPI – RITVA PRÄTTÄLÄ – ERKKI VARTIAINEN – ANTTI UUTELA, Does a Slump Really Make You Thinner? Finnish Micro-level evidence 1978-2002. 01.09.2004. 20 p.

No 929 ANTTI KAUHANEN – HANNU PIEKKOLA, What Makes Performance-Related Pay Schemes Work? Finnish Evidence. 13.09.2004. 22 p.

No 930 KARI E.O. ALHO, Palkankorotusten vaihtoehdot ja talouden kasvu – toimialatarkasteluja Suomen teollisuudella. 04.10.2004. 17 s.

No 931 EDVARD JOHANSSON – HANNU ALHO – URPO KIISKINEN – KARI POIKOLAINEN, Abstaining from Alcohol and Labour Market Underpreformance – Have we forgotten the "dry" alcoholics? 23.09.2004. 12 p.

No 932 MATTHIAS DESCHRYVERE, Health and Retirement Decisions: An Update of the Literature. 24.09.2004. 35 p.

No 933 MATTHIAS DESCHRYVERE, Labour Force Behaviour of Elderly Two Adult Households: Evidence from EU-countries. 24.09.2004. 50 p.

No 934 JARI HYVÄRINEN, EU Outsourcing to the East, Governance and Innovation Systems in the Baltic Countries – A Three-Stage Approach. 27.09.2004. 70 p.

No 935 VILLE KAITILA, Integration and Conditional Convergence in the Enlarged EU Area. 07.10.2004. 17 p.

No 936 MIKA PAJARINEN – PEKKA YLÄ-ANTTILA, Miksi yritykset investoivat ulkomaille? Yrityskyselyyn perustuva analyysi. 08.10.2004. 23 s.

No 937 OTTO TOIVANEN, Choosing Standards. 11.10.2004. 31 p.

No 938 TERTTU LUUKKONEN, Patentointi biotekniikka-alalla. 15.10.2004. 14 s.

No 939 ESSI EEROLA – NIKU MÄÄTTÄNEN – PANU POUTVAARA, Citizens Should Vote on Secession. 20.10.2004. 15 p.

No 940 MIKA WIDGRÉN – PEKKA SULAMAA, Aasian regionalismi vs. globaali vapaakauppa: Simulointitutkimus taloudellisista vaikutuksista. 21.10.2004. 23 s.

No 941 MIKA WIDGRÉN, Suomen, Aasian ja uusien EU-maiden suhteellinen etu ja kilpailuasetelmien muutos. 21.10.2004. 28 s.

No 942 MIKA MALIRANTA – SATU NURMI, Do Foreign Players Change the Nature of the Game among Local Entrepreneurs? 26.10.2004. 22 p.

No 943 JYRKI ALI-YRKKÖ, Impact of Public R&D Financing on Private R&D – Does Financial Constraint Matter? 29.10.2004. 22 p.

No 944 STEFAN NAPEL – MIKA WIDGRÉN, The Inter-Institutional Distribution of Power in EU Codecision. 05.11.2004. 26 p.

No 945 MATHIAS CALONIUS, Luovan yhteiskunnan rakenteet, luovat toimialat ja muotoiluala. 08.11.2004. 31 s.

No 946 ARI HYYTINEN – MIKA PAJARINEN, Is the Cost of Debt Capital Higher for Younger Firms? 11.11.2004. 17 p.

No 947 MARKKU STENBORG, Explaining Open Source. 11.11.2004. 34 p.