Analyse & Kritik 26/2004 (© Lucius & Lucius, Stuttgart) p. 279–301

Margit Osterloh/Sandra Rota

Trust and Community in Open Source Software Production*

Abstract: Open source software production is a successful new innovation model which disproves that only private ownership of intellectual property rights fosters innovations. It is analyzed here under which conditions the open source model may be successful in general. We show that a complex interplay of situational, motivational, and institutional factors have to be taken into account to understand how to manage the 'tragedy of the commons' as well as the 'tragedy of the anticommons'. It is argued that the success of this new innovation model is greatly facilitated by a well balanced portfolio of intrinsic and extrinsic motivation, low costs for contributors and governance mechanisms that do not crowd out intrinsic motivation.

1. Introduction

Until recently it was undisputed that economic development is the more successful

- the more extensive the privatisation of production;
- the more strongly the state is willing to protect private property rights

(e.g. North 1981). A powerful justification for the privatisation of common goods and strong private property rights was given by Garret Hardin's (1968) metaphor of the 'tragedy of the commons'. This metaphor highlights the problems of overuse and undersupply of common resources. Under certain conditions it might be necessary to change the metaphor to 'tragedy of the anti-commons' (Michelman 1982; Heller 1998), in which scattered owners have the right to exclude others from a scarce resource while no one has an efficient right to use it. If the transaction costs to bundle the property rights are too high, such a resource is prone to underuse. This problem might even be worse in the case of *intellectual* property rights, e.g. in biomedical research (Heller/Eisenberg 1998) and the software industry. This was empirically tested in a natural experiment (Bessen/Maskin 2000). Before 1980, patent protection for software was very limited in the United States, as it still is in the EU today. A series of court

* The authors thank Bruce Ackerman, Charles Baden Fuller, Yochai Benkler, Geoffrey Brennann, Bruno S. Frey, Stefan Haefliger, Cynthia Horne, Janos Kornai, Bernhard Kuster, Margaret Levi, Roger Luethi, Michael Baurmann, Susan Rose-Ackerman, Bo Rothstein, Sebastian Spaeth and Georg von Krogh for valuable contributions.

decisions in the early 1980's extended patent protection considerably. Consequently the number of issued patents increased. However, in contrast to what was expected, R&D expenditures relative to sales in relevant samples of the software and software related industries dropped significantly. The question arises under which conditions private ownership of intellectual property rights in effect hinders innovation and when state regulators should be careful in proliferating patent rights.

Employing the example of one of the most innovative industries, open source software production, we study these conditions. Open source software production is an innovation model which is characterized by

1. (partly) public ownership of intellectual property, and
2. user driven distributed innovation.

Some projects like e.g. Linux and Apache managed to attract huge communities of contributors in which intrinsically and extrinsically motivated members voluntarily work together in a complementary manner.[1] We will argue that since open source projects have no clear group and resource boundaries, nor does there exist a central formal authority, the development and maintenance of such communities depends on their ability to

a) develop and enforce rules of cooperation in a self-organized manner, and to

b) develop self-enforcing swift trust which is based on generalized reciprocity between group members.

Open source software is the best known but by far not the only example of this innovation model. Other examples are the NASA Clickworkers (a project where volunteers mark and classify craters on maps of Mars), Slashdot (a site with 'News for Nerds' where users can post submissions, comment on their content and classify the comments themselves as to their helpfulness) and Project Gutenberg (peer-based distribution of books that includes volunteer scanning of hard copies and proofreading) (Benkler 2002). Studying open source software helps us to understand why and when private ownership of intellectual property rights should be proliferated carefully. Even though the creation of rules in open source communities is largely self-organized, state regulators are heavily involved. Thus the endeavour of this paper is threefold. *Firstly*, it studies how trust is developed and sustained in such virtual communities of innovation. *Secondly*, we analyze under what conditions intellectual communities are able to develop and put through their own governance rules. *Thirdly,* it identifies conditions under which private intellectual property rights can hinder innovations. We show that state-imposed private property rights can dramatically impede this process. State intervention thus may solve one tragedy but cause another.

[1] Unfortunately not all projects are thus successful. As an empirical study by Krishnamurthy 2002 shows, many projects indeed dismally fail in this endeavour.

In the *second* section of this paper, a short overview of the characteristics of open source software is provided. The *third* section distinguishes various types of actors in open source software production according to their motivation to contribute to this kind of software. The *fourth* section discusses the role of trust in overcoming a first and second order social dilemma arising in situations of public good production. We argue that only projects that can be trusted to be able to solve both the first and second order social dilemma are attractive for potential new members and can thus hope to attract a large community. We show that on both levels of the social dilemma, trust based on encapsulated interests (Hardin 2002) is not sufficient, but that the emergence of swift trust based on the existence of a sufficient number of intrinsically motivated contributors is an important condition for the success of such projects. In section *five,* we analyse under what conditions this is possible. We show that low cost situations and appropriate institutional arrangements that do not destroy intrinsic motivation are essential for building trust that makes "virtual communities of innovation" work without central authorities and privatisation of intellectual property rights, even when no clear group and resource boundaries exist. We then go on to show how state regulation might adversely affect the open source innovation model by turning low cost into high cost situations. It is *concluded* that considering the complex interplay of motivational, situational, institutional and regulatory factors, trust and motivation issues should be given more weight in designing property rights. Managers and policy makers should be aware that the best policy not always is to blindly apply orthodox economics. Rather, they should consider the variety and interplay of existing extrinsic and intrinsic motivations and establish conditions under which self-governed 'communities of innovation' based on trust can emerge and be sustained.

2. What is Open Source Software?

Open source is a collective term for software licences that give the user the right to read the source code of the software. Users are also allowed to change the source code and to publish these amendments with the original or the changed source code. Furthermore, one is not allowed to raise any licence fees or other fees for the source code. The open source software code thus constitutes a public good in the classical sense. Linux, Apache and Sendmail are three of the most famous examples of this very successful innovation model. Linux as a server operating environment already holds 13.7% of the $50.9 billion market for server computers. This share is expected to rise even further during the next years (Businessweek 2003). In January 2003 the open source web server Apache was used by over 65% of active servers across all domains. It received many industry awards for excellence.[2] Sendmail routes at least 42% of mails in the Internet. In comparison, Microsoft as the closest competitor only holds a market

[2] http://www.netcraft.co.uk/Survey/

share of 18%.[3] SourceForge.net, a repository of open source projects, lists more than 50.000 projects and more than 550.000 registered users.[4]

To explain the success of the new innovation model, one must take into consideration three interlinked characteristics that ensure an efficient concurrence of design and testing in open source software production. These characteristics, which make open source programs more innovative and robust than proprietary programs (Kogut/Metiu 2001), will be discussed in turn:

a) Open source software is produced under licences that assure (partly) *public ownership* by allowing:

- to read and have access to the software's source code as a necessary first step before one can change it,
- to make copies and to distribute those copies,
- to modify the program and distribute modified versions. In the special case of the GNU General Public License, the modified versions have to be published under the same terms as the original software (Stallman 1999).

The various open source licences differ to the extent to which they allow public property to be mixed with private property rights. One of the most far reaching is the GNU General Public Licence (GPL). It forces every program that contains a free software component to be released in its entirety as free software. In contrast to the conventional copyright, this licence is called "copyleft". It 'infects' the open source software with a 'virus' to enforce compliance to the copyleft. Thus, it is ensured that any derived software will remain a public good. Other licenses, like Apache's, allow programmers to make their modifications private and distribute them as proprietary products. This blending of open and proprietary source, however, is sometimes condemned as a threat to the ideals of the open source community (Stallman 1999).

b) *User driven distributed knowledge production* in rapid feedback cycles implements concurrence in design and testing of software modules and thus enables a very efficient new product development process. In traditional software production, the software is usually sold in a form giving no access to the source code. Customers therefore have only limited possibilities to detect mistakes ('debugging') and to improve the program. They can only give feedback to the seller about any malfunctions. In contrast, in open source software production, program innovations are disclosed to the users. A large audience tests the program, debugs it during use and gives immediate feedback. This is the reason why open source software is considered to have a lower defect density than proprietary software: "given enough eyeballs, all bugs are shallow" (Raymond 2001). The user driven rapid feedback cycles work not only with debugging but also with the production of whole modules. These are contributions to the source code which are published and reviewed by peers before they become part of the next release of the software.

[3] http://www.dwheeler.com/oss_fs_why.html
[4] http://sourceforge.net

c) Successful open source projects form *voluntary 'virtual communities of innovation'*.[5]. Following Tönnies (1920) communities are defined as groups of people whose actions are geared towards a collective goal. This differentiates communities from societies, which are abstract entities and in which interactions are characterized by the rational pursuit of individualized goals. In contrast, membership in a community is based on a feeling of belonging and shared values. This *firstly* ensures intensive network embeddedness based on a strong common culture. *Secondly* the members of these communities have a common know how on an expert level without having face-to-face interaction. In contrast to proprietary software, the users of open source software are often more sophisticated. *Thirdly,* Benkler (2002) argues that in virtual communities of innovation, transaction costs of matching talents to tasks can be reduced dramatically compared to market or hierarchical modes of organization. Individuals can judge for themselves in which tasks their talents might be put to most efficient use. This voluntary matching not only avoids information losses which are characteristic for market transactions and within firms, due to incomplete contracts. It also strengthens intrinsic motivation as a result of the autonomy of individuals.[6]

3. Multiple Types of Contributors to Open Source Software

Why should thousands of programmers contribute freely to the provision of a public good? Two alternative explanations are discussed.

- Is it the result of the collaboration of self-interested individuals who invest in their reputation (e.g. Lerner/Tirole 2002a) or who calculate their investment in the open source community lower than their personal benefits (e.g. von Hippel 2001; von Hippel/von Krogh forthcoming)?
- Is it fun, altruism or prosocial preferences which guide the contributors, as some of the leaders of the open source community claim (e.g. Kollock 1999; Raymond 2001; Stallman 1999; Torvalds 1998)?

The two alternative explanations refer to a distinction between two kinds of motivation (Deci/Ryan 2000; Frey 1997; Osterloh/Frey 2000): *Extrinsic* motivation works through indirect satisfaction of needs, most importantly through monetary compensation. *Intrinsic* motivation works through immediate need satisfaction. An activity is valued for its own sake and appears to be self-sustained. The ideal incentive system for intrinsic motivation consists in the work contents itself.

Intrinsic motivation has two dimensions. Following Lindenberg (2001), one can differentiate between enjoyment-based and obligation-based intrinsic motivation.

---

[5] Communities of innovation are comparable to communities of practice. These are groups of people informally bound together by shared expertise and interest (Brown/Duguid 1991; 1998). *Virtual* communities of practice are described by Faraj/Wasko 2001 and Tuomi 2000.

[6] See section 3 on intrinsic and extrinsic motivation.

- *Enjoyment-based* intrinsic motivation is the incentive focused on by Deci and his group (Deci et al. 1999). It refers to a satisfying flow of activity (e.g. Csikszentmihalyi 1975) such as playing a game or reading a novel for pleasure.
- *Obligation-based* intrinsic motivation was introduced by Frey (1997) as a further important form of incentives. Empirical field evidence for the relevance of obligation-based rules are tax morale and environmental ethics, or organizational citizenship behavior (e.g. Organ 1995).

We will argue that in the open source community there exist a variety of ideal types of contributors with different extrinsic and intrinsic motives.[7] In this section, we will distinguish five different types of contributors according to their motives. In reality, these types are overlapping (Hars/Ou 2002; Lakhani et al. 2002).

3.1 Commercial Service Providers

Commercial service providers make money with open source software in spite of the fact that open source software is a public good. The most prominent example is Red Hat. This company does not actually sell the source code (which anybody can download from the Internet for free). Instead, it sells support and services. In addition, it adds value by integrating autonomous open source components into a working and reliable operating system that can easily be installed by inexperienced users. Other commercial firms like Hewlett Packard sell hardware, like printers, and contribute add-ons, like printer drivers, to make their products work with open source software. Finally, companies like IBM contribute to open source software by making their hardware compatible with it. These firms are absolutely vital for the widespread adoption of this kind of software, because the inexperienced consumer gets reliable services and add-ons, thus helping to drive these programs into the mainstream (Kogut/Metiu 2000).

3.2 Software Customizers

Contributors to open source can gain non-monetary benefits by tailoring the software to their own needs (von Hippel 1988; von Hippel/von Krogh forthcoming). They follow the saying "if you want something done right, do it yourself" (Lakhani/von Hippel forthcoming). They have sufficient incentives to contribute to an innovation when they expect the personal benefits to exceed their costs.

Why should the benefits of publishing one's improvements on the Internet exceed the costs of revealing information? It is argued that, *firstly,* publication opens up the possibility that other users might work with the amendments of the code, maintain and develop them. That includes the elimination of possible errors (e.g. von Hippel 2001; Lerner/Tirole 2002a). *Secondly,* the Internet makes it possible for a software developer to access a wide audience with very low costs.

[7] An ideal type, according to Weber 1949, is a construct which has been arrived at by the thinking accentuation of certain elements of reality. In its conceptual purity, this ideal type cannot be found empirically anywhere in reality.

Because publication costs are small, publication on the Internet can pay off even if the expectations for helpful comments from other users are relatively low. Besides, the gains the developer reaps from the newly developed functionalities are not diminished by additional users.

3.3 Reputation Investors

Contributors can make money *indirectly* by signalling their ability in the open source community which can then be turned into money through employment by a commercial software company or through easier access to venture capital. Employers or venture capitalists can take the reputation of a programmer as a signal for his/her abilities which would otherwise be hard to identify. It is argued that in open source projects reputation can be more easily made visible than in proprietary projects due to the system of files that list people who made contributions, and due to the public nature of mailing list archives (Lerner/Tirole 2002a; Moon/Sproull 2000). This system makes open source production comparable to the production of research in an academic community where reputation is made visible through citations. As in the academic community, strong norms exist regarding the public validation of innovative results.

3.4. Homo Ludens

While commercial service providers, software customizers and reputation investors are extrinsically motivated, much evidence exists that for many programmers the work itself is intrinsically rewarding. This idea corresponds to Huizinga's (1986) *homo ludens*, the playful human that receives some form of benefit simply from carrying out the programming or from dealing with a software problem. In that case, contributions to the open source code are not a cost but a benefit, not investment but consumption. Important contributors to open source software report that they are doing the programming "just for fun" and the public display of one's abilities (Torvalds/Diamond 2001).[8] As Raymond (2001) puts it: "We're proving not only that we can do better software, but that joy is an asset" (see also Brooks 1995). Writing or debugging software is perceived as a "flow experience" (Csikszentmihalyi 1975). More than 70% of open source developers report that they lose track of time while programming (Lakhani et al. 2002). As Ullman (1997) shows, programmers often experience a strong personal satisfaction from creating 'something that works'. This kind of motivation is fostered by voluntary work without time pressure (Deci et al. 1999). Creativity and motivation to volunteer in unpaid helping activities are higher when the external pressure is low (Stukas et al. 1999; Amabile et al. 2002). Not being subjected to delivery deadlines is an important characteristic of open source projects (Raymond 2001).

[8] Displaying ones abilities is intrinsically motivated in so far as it is not aimed at building monetarizable reputation but rather at forming a consistent self-image or, as Akerlof and Kranton 2000 call it, a feeling of identity.

3.5 Members of the Tribe

The open source community is often described as a gift-culture instead of an exchange culture (e.g. Raymond 2001). A gift is characterized by receiving no tangible rewards but psychological benefits such as the 'warm glow' of sympathy or the satisfaction of living up to a moral commitment (Rose-Ackerman 1998). Gift-giving reveals the motivations of altruism or generalized reciprocity. Open source contributors report that they like the sense of 'helping others' or 'giving something back' to like-minded others (Faraj/Wasko 2001). Norms of generalized reciprocity sustain kindness as a social institution and lead people to provide help (Constant et al. 1996). These motivations are apart from transactional exchange relationships, because the receiver is often unknown to the giver. Participants report that "the person I help may never be in the position to help me, but someone else might be" (Rheingold 1993). People seem to reply to the entire group when answering an individual question (Wellman/Gulia 1999). The good of the community enters into the preferences of the individual contributor. Thus a reciprocal trust based on shared values and on warm feelings towards the group is existing (Rose-Ackerman 2001), rather than trust based on encapsulated interests (Hardin 2002).

The belief that it is the right thing to give software away as a common good leads to the corollary that private ownership of intellectual property can be damaging.[9] The open source movement seems to be fuelled to some extent by the aim to destroy Microsoft's monopoly (e.g. Markus et al. 2000; Raymond 2001). Members of the tribe thus produce a public good of two different orders. *Firstly*, they contribute to the functionality and quality of the programs (first order public good). *Secondly*, they are engaged in monitoring and sanctioning activities to ensure that the source code stays open (second order public good[10]). This includes a heated discussion between various fragments of the open source community on what kind of licence best supports these

moral concerns. While some believe that only the GNU General Public Licence guarantees that source code remains open, others feel that the viral effect of this licence actually reduces freedom.

---

[9] Empirical evidence about the importance of this motive within the open source community is ambivalent. According to Ghosh et al. 38% of open source developers report that their motivation to contribute to the community is their believing in that software should not be a proprietary good (Ghosh et al. 2002). In a different survey, Lakhani et al. 2002 find that 11% of open source developers are driven by the motivation to beat proprietary software. The difference might be explained by different samples and methodology. As always one has to be careful to take data based on self-reports at face-value due to possible social desirability biases. However, as heated discussions on the Internet about license terms and the examples discussed in chapter 5.1 show, a significant part of the community demonstrates their concerns about keeping the source code open very actively.

[10] For first and second order public goods see section 4.

3.6 Complementarity of the Different Types

We showed that in the open source community different types of motivation among the contributors exist. In the following we argue that these different types do not only coexist but are complementary to each other.

Intrinsically motivated 'fun seekers' and 'members of the tribe' play an especially important role during the starting phase of open source projects. In this phase a usable product does not yet exist and the risk of project failure is very high. The enthusiasm of intrinsically motivated contributors helps these projects to gain enough momentum. In that way, they reduce the costs for extrinsically motivated contributors so that contribution becomes more attractive for them (Bessen 2002; Franck/Jungwirth 2003). Without intrinsically motivated contributors

- commercial service providers would lack the basis of their business. They make money on support only if the open source software is successful.
- software customizers would have to make higher set up investments so that the costs are more likely to exceed the benefits.
- reputation investors would not be able to produce marketable signals. Employers and venture capitalists are only attracted by successful projects that have already produced a critical mass of source code.[11]

On the other hand the success of open source software is dependent on extrinsically motivated contributors. Commercial players, software customizers or reputation investors trigger a leverage effect:

- If the open source movement were solely based on intrinsic motivation, the products would not be linked the way they are to the needs of the users. A disadvantage over commercial development might result.
- Inexperienced consumers could not use this kind of software which was originally designed by and for experts.

So far, it is not known which proportion of intrinsically and extrinsically motivated people exist. But there are some preliminary empirical findings about what presently drives open source programmers and participants of newsgroups. In an empirical study with participants of a user-to-user Apache field support system Lakhani and von Hippel (forthcoming) report generalized reciprocity as the most agreed-with statement ("I have been helped before, so I reciprocate", "I help now so I will be helped in the future"), followed by identification with

[11] The decision of reputation investors to join a project in its starting phase depends on their attitude towards risk taking. While risk adverse reputation investors will wait and see how a project develops before deciding to join, reputation investors with risk seeking preferences might be willing to join before it is clear whether the project will be a success or a failure. In both cases, however, the risk that the reputation investor has to take is lower the higher the number of intrinsically motivated contributors. We thank an anonymous referee for help in clarifying this point.

the community ("I answer to promote open source software"). These self-reports might emphasize 'socially correct' answers. Observable behaviour however supports these claims. The same empirical study found that in one of the Usenet newsgroups 57% asked questions only (and can thus be classified as free riders), 21% both asked questions and gave answers (they can be classified as reciprocators), and 22% provided answers only. Whether they do that to invest in their reputation or for altruistic reasons cannot be established by the data. Taking into account other empirical work one may conclude that extrinsic and intrinsic motivation exert about the same influence (Ghosh et al. 2002; Hars/Ou 2002).

4. The Role of Trust in Open Source Communities

In open source extrinsically and intrinsically motivated contributors work together in a complementary manner to produce a public good. In the absence of a central authority that has the power to enforce contribution, the presence of extrinsically motivated utility-maximizers usually leads to an undersupply and overuse of public goods due to free-riding. This problem is known in the literature as the social dilemma: Social dilemmas arise if the actions of self-interested individuals do not lead to socially desirable outcomes (Dawes 1980; Ostrom 1998). The consequence might be a 'tragedy of the commons' as Hardin (1968) described it, where the public good is not produced at all. But why do we not observe this problem in open source? In this chapter we turn to the role of trust and intrinsic motivation in overcoming the social dilemma in open source.

In open source, the social dilemma is located on different levels. *On the first level* free riding can take place with respect to the production of software itself, because open source software constitutes a public good. Since nobody can be excluded from open source software, there is a problem of undersupply.[12]

On the second level, the rules of the game that prevent first order free riding have to be observed and sanctioned. The worst kinds of first order free riding are not honouring the terms of the licence, using open source components in proprietary commercial products without giving anything back to the community or not citing or removing the credits of a contributor. As Raymond (2001) points out "surreptitiously filing someone's name off a project is, in cultural context, one of the ultimate crimes". Second order free riding is related to the monitoring of the compliance to the open source software norms and to the application of sanctions that prevent first order free riding. Reprimanding rule breakers in order to enforce the code of ethics is itself a public good and thus constitutes a social dilemma of a higher order: "Punishment almost invariably is costly to

[12] In newsgroups this kind of free riding is known as lurking. This means reading ongoing discussions without contributing. Lurking is usually not really a problem as long as enough individuals are willing to contribute, because there is no rivalry in consumption. But there can exist a rivalry in attention, due to excessive crossposting and trolling. Given the huge amount of information that is transferred, it is critical that contributors respect the focus of the problem that is dealt with and therefore avoid crossposting (Kollock/Smith 1996). Trolling refers to deliberately posting messages with no other aim than to provoke other users.

the punisher, while the benefits from punishment are diffusely distributed over all members. It is, in fact, a public good" (Elster 1989, 41).

As we will show trust is necessary on both levels to enhance cooperation. But what form does trust in open source communities take? It is important to note that trust in open source communities is rather institutional than personal. The number of participants to a given open source project is often very large and the communities are open to exits and new entries. Thus the development of trust cannot be based on repeated interactions of the same individuals who get to know each other over time and learn to trust each other. Rather, trust development in open source communities takes the form of what Meyerson et al. (1996) termed swift trust.

'Swift trust' describes a form of trust that is found in teams that only work together for a limited period of time and do not have the opportunity to develop trust based on personal relationships and mutual control. Members of such temporary teams decide on how much they think they can trust the others even before actually joining the team. This decision is based on stereotypical social categories and on a subjective appraisal of the intrinsically motivated adherence to mutual norms of reciprocity within a community.

We differentiate between two different types of swift trust: trust based on encapsulated interests (Hardin 2002) and cognitive trust which is based on knowledge about the characteristics of the trustees, e. g. their dominating (intrinsic or extrinsic) motivation (Lewicki/Bunker 1995). *Trust based on encapsulated interests* means that I trust somebody because I believe it is in the trustee's (i.e. the trusted person's) best personal interest not to deceive my trust. This kind of trust is based on an estimation of the situation in which an interaction takes place. Given this specific situation, do I believe that the other person has enough incentives to behave trustworthy? *Cognitive trust*, on the other hand, is based on an estimation of the characteristics of the person I interact with. Do I believe that a person will honour my trust even if it would be in their best personal interest not to do so? In the open source context cognitive trust means trust that there are a sufficient number of intrinsically motivated contributors in a given project. The trustor himself might well be extrinsically motivated, i.e. he might contribute to the open source community in an instrumental way.

The development of trust is especially important for potential new members, because they will only be willing to join a project which they believe is able to solve the first and second order social dilemmas in a sustainable way. We will now discuss the social dilemmas in open source in turn.

As it turns out, the solution of the first order social dilemma is not a big problem in open source. As we showed in chapter 3, contribution to open source projects is not a pure public good. The different types of contributors all have individual incentives to participate which make the contribution option more worthwhile for them than merely free-riding. This holds for extrinsically as well as intrinsically motivated contributors. If enough people with sufficient individual incentives exist, the social dilemma is transformed into a coordination game where more than one equilibrium exists (Sen 1974). In a coordination game, the production of a public good mainly depends on the estimation of potential con-

tributors, how many others will contribute as well. This is equivalent to saying that the contribution decision depends on the amount of trust based on encapsulated interests, i.e. the belief that it is in the personal interest of a sufficient number of other potential contributors to participate.

But even on this level trust based on encapsulated interests alone is not enough. As we showed in chapter 3.6., especially in the beginning phase of a project, success is very much facilitated by a high number of intrinsically motivated participants. This means that also on the level of the first order social dilemma, cognitive trust in the intrinsically motivated trustworthiness of a relevant part of the community is a very important condition.

Potential new members can estimate the trustworthiness of a community simply by observing its behaviour. Due to the publicity of the Internet they are able to judge whether norms of generalized reciprocity are lived up to within a community before deciding to join. For example they can observe whether members offer mutual support, provide helpful remarks to each other and answer questions in newsgroups.

Thus the first order social dilemma in open source can be solved even in the presence of purely extrinsically motivated trustors. Unfortunately, this kind of cooperation is rather unstable. As soon as golden opportunities arise, self-interest maximizing egoists cannot be counted on their cooperation anymore. This is the reason why the enforcement of cooperation rules is so important. They protect the community from exploitation by opportunists who face a golden opportunity.

Second order social dilemmas of rule enforcement can be solved without a central authority if a sufficient number of obligation-based intrinsically motivated people exist who are prepared to punish rule-breakers even if such punishment is costly to them. Laboratory empirical evidence for the existence of such people can be found in one-shot public good games (Camerer/Fehr forthcoming; Ledyard 1995). In the open source community, these sanctions take place by violently blaming individuals on the Internet, called 'flaming'. Flaming is not simply a way of punishing rule-breakers, but also has an expressive function in assuring users that others are doing their part in using the public good wisely (Kollock/Smith 1996).[13] Other sanctions are the public announcement of 'killfiling' (stating that one doesn't want to receive mails from a specific person) or shunning (deliberately refusing to respond).

Monitoring the behavior of participants is often easy in the open source community because the Internet gives full transparency.[14] Sanctioning, however, is more of a challenge. *Firstly*, many sanctions (like flaming) are informal in nature. *Secondly*, the community members are often anonymous and no clear group and resource boundaries exist. Insofar the conditions in open source communities are different from the communities Ostrom (1990) has analysed. She argues that only if clearly defined group and resource boundaries exist, self governance of

[13] This is especially important for potential new contributors who, before deciding to contribute, need to estimate whether they can trust in the stability of a community.

[14] With the exception of illegally including open source code in proprietary programs.

the commons can be successful. Nevertheless in the open source community self governance works. It is reported that sanctions have a significant effect on behaviour (Kollock/Smith 1996) though these sanctions often do no actual harm. They can influence behaviour indirectly however by damaging the sanctioned member's reputation or by inducing shame. In these cases, one has to assume that not only the sanctioner, but also the sanctioned person must be intrinsically committed to obligation-based rules. Purely extrinsically motivated egoists would not feel any shame (Elster 1999; Orr 2001). It can be concluded that the solution of the second order social dilemma in open source software production is greatly facilitated by the presence of intrinsically motivated trustees.

Again, potential new members judge the trustworthiness of a community by observing its behaviour. If the existence of intrinsic motivation facilitates the development of swift trust and the complementary interaction of the different types of contributors to open source software, the question arises, under which conditions the required amount of intrinsic motivation and trust can exist and be maintained.

5. The Antecedents of Swift Trust in Open Source Projects

We showed that different kinds of trust are needed on the trustor's and the trustee's side. On the trustor's side, extrinsically motivated trust based on encapsulated interests is sufficient. In contrast, on the trustee's side, there must exist a sufficient number of intrinsically motivated contributors as a precondition for the development of swift trust on the trustor's side. The problem therefore boils down to the question, under what conditions the trustor will judge the probability that a sufficient amount of intrinsic motivation exists and can be maintained as high. We identify two antecedents: *Firstly*, institutions must be created that ensure that the existing intrinsic motivation of the trustees is not destroyed. *Secondly*, even intrinsically motivated members will not contribute to a public good if the costs of doing so are high. Thus keeping the contribution costs low is the second antecedent of swift trust. We then go on to show that state intervention can disturb the equilibrium between intrinsically and extrinsically motivated contributors by turning low cost into high cost situations.

5.1 The Institutional Level: Initial Intrinsic Motivation Must Not be Crowded Out

It is hard to analyze the reasons why people develop an initial sense of fun for or a commitment to certain projects. But we know the institutional conditions under which initial intrinsic motivation is crowded out (undermined) or crowded in (strengthened) by external interventions (Frey/Osterloh 2002). Two conditions are relevant for the required institutional governance mechanisms to foster the new innovation model: self-determination (1) and conditional cooperation (2).

1) *Self-determination*: External interventions crowd out intrinsic motivation if the individuals affected perceive them to be controlling. In that case self-determination and self-esteem suffer and the individuals react by shifting their

'locus of causality' from inside to outside. In contrast, intrinsic motivation is crowded in if a person's feelings of self-determination are enhanced. (Deci/Ryan 2000; for a comprehensive overview over the empirical evidence see Frey/Jegen 2001).

In open source projects self determination is enhanced for *two* reasons. *Firstly*, contributors choose for themselves where and what they wish to contribute (Benkler 2002; for empirical evidence see von Krogh et al. 2002). *Secondly*, a variety of self governance mechanisms give contributors large possibilities to participate in collective decision making in a transparent way. Extensive experimental and field research show that civic virtues are strengthened by procedural utility (Benz et al. forthcoming; Frey/Stutzer 2002; Osterloh et al. 2002) and that 'organizational citizenship behavior' is strengthened by participation and procedural fairness (e.g. Organ/Ryan 1995). Though governance rules in open source projects differ to a great extent (for an overview see Markus et al. 2000), open source contributors submit themselves voluntarily to these rules without any contract.

2) *Conditional cooperation*: Empirical evidence shows that many individuals contribute voluntarily to public goods in social dilemmas as long as some other individuals contribute also. They are conditional cooperators (Fischbacher et al. 2001; Levi 1988; Ostrom 2000). KDE (K Desktop Environment) gives an impressive example of how conditional cooperation can be undermined. KDE is a Windows-like desktop for Linux and other open source operating systems developed by the open source community. It is based on a graphical interface toolkit called Qt. Qt was developed by Trolltech, a commercial software firm. It did not comply with the requirements of open source (Stallman 1999). Even though the Linux community agreed that KDE was a technically excellent product, many members refused to endorse it because they didn't agree with the terms of the Qt licence. These members started a parallel project called GNOME that is distributed under copyleft. Finally Trolltech reluctantly relicensed their product. By now Qt is available under a copyleft license.

Two consequences to maintain conditional cooperation follow. *Firstly,* intrinsic motivation is crowded out by free riders. Therefore institutional governance mechanisms must be set in place which hinder exploitation of voluntary donors. Open source licences, in particular copyleft, are such institutional mechanisms (Franck/Jungwirth 2003). They impose to contributors as well as to free riders what Hansmann (1980) calls the nondistribution constraint which is characteristic for non profit organizations (Rose-Ackerman 1996): Voluntary contributions cannot be redistributed among those who have a main impact on the organization. The nondistribution constraint is a major institutional precondition for voluntary donations to organizations. It is the reason why institutions like the Red Cross or most universities are governed as non profit organizations. Also for commercial providers who are dependent on the goodwill of the developers, it is crucial to commit credibly to the nondistribution constraint and not to appropriate the joint project in an unfair manner (Franck/Jungwirth 2003). Otherwise conditional cooperation breaks down and their business model will fail (Benkler 2002).

Red Hat has submitted itself voluntarily to constraints beyond the obligations of the open source license to strengthen conditional cooperation: As mentioned, Red Hat does not sell the Linux code. Instead, it sells support, services and value added by assembling and testing a running operating system that is compatible with other products carrying the same brand. After a short while, other CD-ROM distributors were advertising the same CD-ROM for a considerably lower price than Red Hat charged for its product. Even if Red Hat does not own intellectual property rights on the entire source code distributed on their CD-ROM, they do have the copyright on parts of the CD. How did Red Hat react? The somewhat astonishing answer is: not at all. The managers of Red Hat argued that the norms of the open source community precluded any claim on private property rights on their product. Since Red Hat is dependent on the goodwill of the open source community, it adheres to rules which foster conditional cooperation.

Secondly, as experiments about conditional cooperation show, intrinsic motivation is crowded out if the existing rules of cooperation in a public good situation, e.g. the rules of nondistribution, are disregarded (Fischbacher et al. 2001). Therefore rule breaking must be hindered. Rule breaking can be made more difficult if costs for monitoring and sanctioning are low. As mentioned, monitoring in open source software production is easy due to the publicity of the Internet. Also, sanctioning by flaming or kill-filing is a low cost activity. But it seems reasonable to assume that in particular anonymous defectors are not vulnerable by sanctions. In these cases, sanctioning is only insofar effective as defectors in the open source community still feel a minimum of intrinsically motivated shame. Because informal graduated sanctions have a strong expressive function, they are very suitable not to crowd out shame by alienating people from the community (Kollock/Smith 1996; Orr 2001; Ostrom 1990).

5.2 Low Cost Situations: Economy of Virtue

Why does production flourish in open source projects based on the intrinsic motivation of many people? Why do other fields like the pharmaceutical or bio-medical industry not apply this model? The simple answer is that this production model only works in situations in which the benefits exceed the costs. This is also true for intrinsic benefits. Even among intrinsically motivated donators, martyrs and saints are in short supply. Donators are more willing to contribute if the private opportunity costs are not too high (Rose-Ackerman 2001, 553), thus 'economizing on virtue' (Ackerman 1993, 198). According to North (1990, 43) there is a downward sloping demand curve for moral concerns. The more costly it gets, the less people contribute. On the other hand, if there exists a low cost situation, many people contribute small bits to the public good so that the total amount of contributions rises considerably (Kirchgässner 1992; Kliemt 1982).

We distinguish *two* different aspects: The costs and benefits of actually producing the code and the costs and benefits of revealing it to the community. Even though these aspects are intermingled, we will now consider them in turn.

1) For the actual *production of the source code*, two conditions which are often found in system product industries are beneficial for the cost/benefit ratio, namely sequential and complementary innovation processes (Bessen/Maskin 2000; Somaya 2003).

- *Sequential* means that each successive invention builds on the preceding one, in particular that there are incremental, not radical steps of innovation. This allows users and contributors to amortize their initial investments in project specific human capital over several rounds of innovation, thus keeping costs low.[15]

- Innovation is *complementary* if several inventors, by following somewhat different research lines, enhance the overall probability that a certain problem is solved or, more generally, an innovation arises. This can be explained by internal dynamic economies of scales alongside trajectories, which help to concentrate successful search to a narrow field. Thus, if the efforts of an actor further the chances of discovery sufficiently, potential benefits are enhanced and this actor has an incentive to contribute to the process of discovery even if, in principle, she could wait until someone else makes the invention. The impact of such trajectories is of greater weight during the period of exploration than during exploitation (March 1991), because in that period uncertainty is more important.

2) Software is an area where the monetary *costs of revealing innovations* compared to the benefits are often quite low. There are *two* kinds of monetary costs to be considered. *Firstly*, costs of diffusion are low. Participants simply post their contributions on the appropriate Internet site. *Secondly*, the losses stemming from sharing intellectual property rights by using an open source license are often low compared to the gains from the expected feedback by other participants.

In low cost situations, chances that many people are sufficiently intrinsically motivated to contribute their bit to the first and second order public goods are higher. Trustors have more reason to believe that the social dilemmas on both levels will be solved within a community. In the next section we show that state intervention may well turn low into high cost situations, thus undermining the necessary antecedents of both trust based on encapsulated interests and cognitive trust in the intrinsically motivated trustworthiness of a sufficient number of contributors.

5.3 The Impact of State Regulation: Turning Low Cost into High Cost Situations

As we showed, open source software production challenges conventional economists' wisdom that innovations are better supported the more they are protected by private intellectual property rights (e.g. North 1981). We showed that in

[15] Von Krogh et al. 2002 studied the process of joining an open source project empirically, using the example of Freenet. They show that this joining process takes quite some time.

many open source projects the problem of underprovision seems to have been overcome by a complex interplay of extrinsically and intrinsically motivated contributors.[16] This interplay depends on the one hand on institutions that do not crowd out intrinsic motivation and on the other hand on low cost situations which keep the costs of intrinsically motivated moral behaviour within reasonable limits. However, state intervention could very well damage the future success of this new innovation model.

Software production is an example of collective production in which private property rights might even cause a 'tragedy of the anti-commons' (Heller 1998; Heller/Eisenberg 1998). If property rights on a resource are scattered among many parties and the transaction costs to bundle the property rights are too high, this resource is prone to underuse.[17] In the case of intellectual property rights exclusion is made possible by patents. While patents are intended to further innovation by enabling innovators to collect the rents on their investment, in some cases they are used solely to block competitors from using an innovation.

New software is usually based on existing programs and develops them further. Some authors even go as far as to state that software development today is impossible without infringement of intellectual property held by some other party (Bessen 2002).

For open source software production, the potential effects of patent protection are even worse than in proprietary software production for several reasons. *Firstly*, since open source software developers cannot make money with their programs as such, they simply cannot afford to stand up in legal fights about patent infringements (Bessen 2002).[18] *Secondly*, especially in systems product industries like the computer and software industry, patents can be used to strengthen one's bargaining power in the case that some other party wants to block access to its own patents. Systems product industries are characterized by the fact that their products incorporate numerous inventions made by other parties. Since access to others' patents is essential, firms can build up patent portfolios to have something valuable to offer in exchange. Somaya (2003) tested this empirically and found evidence for this behaviour in the computer industry. In open source projects by definition it is impossible to build up such bargaining power. If a patent holder chooses to sue for infringement, open source projects have nothing to offer in exchange for an out of court settlement. The resulting 'patent thickets' threaten the ability of open source developers to improve software (Bessen 2002). *Thirdly*, the option to patent software rather than simply

[16] In open source software production, the problem of overuse does not occur since there is no rivalry in consumption. Additional users can even generate positive external network effects.

[17] An example of how privatisation in postsocialist economies can trigger a 'tragedy of the anti-commons' was given by Heller 1998. He started by asking why several years after the transition 'from Marx to markets' storefronts often remain empty while small kiosks full of goods mushroom on the streets. He argues that the problem is not primarily a lack of clearly defined property rights, corruption, or disobedience of the law, but the way government scatters property rights rather than creating coherent bundles of rights.

[18] So far only a few open source developers have been sued for patent infringement (Bessen 2002). Still the potential threat must not be neglected.

having the copyright on the source may simply increases opportunity costs, thus turning low cost into high cost situations.

When faced with situations where such a tragedy of the anti-commons could arise, governments should not blindly apply orthodox market economics by increasing the scope and sophistication of regulations for private appropriation of intellectual property rights. Regulators should spend their efforts providing tools which help to avoid the 'tragedy of the anti-commons' rather than supplanting copyright protection, on which the viability of the new innovation model depends, with patent protection (Benkler 2002).

6. Concluding Remarks

Open source software production is a highly successful innovation model. But it is, by far, not a singular case but rather one example of 'virtual communities of innovation'. The purpose of this paper is to inquire under which conditions the new innovation model might work in general.

We argued that a stable solution of the social dilemmas of public good production is greatly facilitated by the existence of different kinds of motivations and trust. The first order public good consists of direct contributions to the development of open source software. The second order public good consists of monitoring and sanctioning deviations from the open source software norms. It is necessary to maintain conditional cooperation and generalized trust of the benevolent contributors. *Extrinsically* motivated participants only contribute to the first order public good in an instrumental way. Their aim is to tailor the products to their own needs, to invest in their reputation for monetary purposes or to enlarge the user base for their complementary products. They can only be trusted to contribute as long as it is in their best personal interest to do so. *Intrinsically* motivated members on the other hand contribute to the first and the second order public good. Intrinsic motivations are twofold, enjoyment-based (fun, public display of ones abilities) and obligation based (following norms of generalized reciprocity). Without these intrinsically motivated contributors cooperation in open source software is not sustainable since in many cases only their efforts safeguard the community against exploitation by purely self-interest maximizing opportunists in the presence of a golden opportunity. Thus, potential new members will only join a community if they can build cognitive trust that there exist a sufficient number of intrinsically motivated members who are willing to contribute to the public goods even if the costs should exceed their personal benefits.

Thus the trustworthiness of a community to a large degree depends on the amount of intrinsic motivation of a relevant part of its members. We identified two conditions that make the presence of this kind of motivation more likely: Institutional arrangements that do not crowd out intrinsic motivation and low cost situations.

On the institutional level, licences like copyleft seem to be a good solution to foster the conditional cooperation of intrinsically motivated contributors on the

one hand and serve the interests of extrinsically motivated investors on the other hand (Franck/Jungwirth 2003). Nevertheless, it seems to be important that the commercial providers commit themselves to the norms of the open source community beyond the obligations of the open source licence. As a consequence, companies like Red Hat submit themselves voluntarily to constraints beyond copyleft to maintain conditional cooperation in the community.

Low cost situations foster trust by making intrinsically motivated contributions to the public goods more likely, thus strengthening the trustworthiness of the community. Thus, situational and motivational factors are highly interlinked.

Low cost situations also explain why private ownership of intellectual property rights is sometimes inefficient. Private ownership of intellectual property among independent suppliers is less efficient in low-cost situations, (a) characterized by incremental and complementary innovations or (b) whenever concurrence of design and testing is crucial. In these cases, owners have a right to exclude others from a scarce resource while no one has an efficient right to use it. This condition not only holds in software production but also in other peer productions of intellectual goods (Benkler 2002). It does not hold in situations (e.g. in the pharmaceutical industry) where testing and market launching demand high investments. Further empirical research is needed to shed light on the quantitative dimension of this situational factor.

Even though this new innovation model is based on self-regulation in the absence of a central authority, state intervention in the form of a strengthening of intellectual property rights could very well damage its success. Patent protection can not only reduce the room in which these self-governed institutions can flourish. They also turn low cost into high cost situations and therefore diminish individual willingness to contribute to the public good.

To conclude, though we are far from fully understanding the interplay between motivational, situational, and institutional factors which make this new innovation model work, open source software production shows that even in virtual communities with no clear cut boundaries, under certain conditions trust and trustworthiness can flourish.

Bibliography

Ackerman, B. (1993), *We the People*, Cambridge/MA

Akerlof, G. A. (1982), Labor Contracts as Partial Gift Exchange, in: *Quarterly Journal of Economics 97*, 543–569

Akerlof, G. A./R. E. Kranton (2000), Economics and Identity, in: *Quarterly Journal of Economics 105*, 715–753

Amabile, T. M/C. N. Hadley/S. J. Kramer (2002), Creativity under the Gun, in: *Harvard Business Review 80*, 52–61

Barbera, F. R. (1999), Berkman Center rebuffed, in: *Harvard Law Record 109*, 1

Benabou, R./J. Tirole (2002), *Intrinsic and Extrinsic Motivation*, Princeton University Working paper, Princeton

Benkler, Y. (2002), Coase's Penguin, or, Linux and the Nature of the Firm, in: *The Yale Law Journal 112*, 369–446

Benz, M./B. S. Frey/A. Stutzer (forthcoming), Introducing Procedural Utility: Not only What, but also How Matters. in: *Journal of Institutional and Theoretical Economics*

Berlecon Research (2002), Floss Final Report – Part 3: Basics of Open Source Software Markets and Business Models, http://www.berlecon.de/studien/floss/FLOSS_Grundlagen.pdf

Bessen, J. (2002), What Good is Free Software?, in: R. W. Hahn (ed.), *Government Policy Toward Open Source Software*, Washington, 12–33

— /E. Maskin (2000), *Sequential Innovation, Patents and Imitation*, MIT Economics Department Working Paper, Cambridge/MA

Brooks, F. P. (1995), *The Mythical Man-Month: Essays on Software Engineering*, Reading

Brown, J. S./P. Duguid (1991), Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation, in: *Organization Science 2*, 40–57

— /P. Duguid (1998), Organizing Knowledge, in: *California Management Review 40*, 90–111

— / — (2001), Knowledge and Organization: A Social-Practice Perspective, in: *Organization Science 12*, 198–213

Businessweek (2003), The Linux Uprising, 3.3.2003

Camerer, C. F./E. Fehr (forthcoming), Measuring Social Norms and Preferences Using Experimental Games: A Guide for Social Scientists, in: J. Henrich/R. Boyd/S. Bowles/C. F. Camerer/E. Fehr/H. Gintis (eds.), *Cooperation, Self Interest and Punishment: Experimental and Ethnographic Evidence from Small-Scale Societies*, Oxford

Constant, D./L. Sproull/S. Kiesler (1996), The Kindness of Strangers: The Usefulness of Electronic Weak Ties for Technical Advice, in: *Organization Science 7*, 119–135

Csikszentmihalyi, M. (1975), *Beyond Boredom and Anxiety*, San Francisco

Dawes, R. M. (1980), Social Dilemmas, in: *Annual Review of Psychology 31*, 169–193

Deci, E. L./R. M. Ryan (2000), The 'What' and 'Why' of Goal Pursuits: Human Needs and the Self-Determination of Behavior, in: *Psychological Inquiry 11*, 227–268

— /R. Koestner/R. M. Ryan (1999), Meta-Analytic Review of Experiments: Examining the Effects of Extrinsic Rewards on Intrinsic Motivation, in: *Psychological Bulletin 125*, 627–668

Dempsey, B. J./D. Weiss/P. Jones/J. Greenberg (2002), Who is an Open Source Software Developer? in: *Communications of the ACM 45*, 67–72

Deutsche Bank Research (2002), Free Software, Big Business? Open Source Software Tightening their Grip on Industry and the Public Sector, in: *E-conomics 32*

Elster, J. (1989), *The Cement of Society: A Study of Social Order*, New York

— (1999), *Alchemies of the Mind: Rationality and the Emotions*, New York

Faraj, S./M. M. Wasko (2001), *The Web of Knowledge: An Investigation of Knowledge Exchange in Networks of Practice*, Florida State University Working Paper, Tallahassee

Fischbacher, U./S. Gächter/E. Fehr (2001), Are People Conditionally Cooperative? Evidence from Public Good Experiments, in: *Economic Letters 71*, 397–404

Franck, E./C. Jungwirth (2003), Reconciling Investors and Donators – The Governance Structure of Open Source, in: *Journal of Management and Governance 7*, 401–421

Frey, B. S. (1997), *Not Just for the Money: An Economic Theory of Personal Motivation*, Cheltenham

— /R. Jegen (2001), Motivation Crowding Theory: A Survey of Empirical Evidence, in: *Journal of Economic Surveys 15*, 5, 589–611

— /M. Osterloh (2002), *Successful Management by Motivation. Balancing Intrinsic and Extrinsic Incentives*, Berlin

— /A. Stutzer (2002), *Beyond Outcomes: Measuring Procedural Utility, Berkeley Olin Program in Law & Economics*, Working Paper Series, Working Paper 63

Ghosh, R. A./R. Glott/B. Krieger/B. Robles (2002), Floss Final Report. Part 4: Survey of Developers, http://www.infonomics.nl/FLOSS/report/FLOSS_Final4.pdf

Hansmann, H. B. (1980), The Role of Nonprofit Enterprise, in: *Yale Law Journal 89*, 835–901

Hardin, G. (1968), The Tragedy of the Commons, in: *Science 162*, 1243–1248

Hardin, R. (2002), *Trust and Trustworthiness*, New York

Hars, A./S. Ou (2002), Working for Free? Motivations for Participating in Open Source Projects, in: *International Journal of Electronic Commerce 6*, 25–39

Heller, M. A. (1998), The Tragedy of the Anticommons: Property in the Transition from Marx to Markets, in: *Harvard Law Review 111*, 622–688

— /R. S. Eisenberg (1998), Can Patents Deter Innovation? The Anticommons in Biomedical Research, in: *Science 280*, 698–701

Huizinga, J. (1986), *Homo Ludens*, Beacon Press

Kirchgässner, G. (1992), Toward a theory of low-cost-decision, in: *European Journal of Political Economy* 8, 2, 305–320

Kliemt, H. (1982), The Veil of Insignificance, in: *European Journal of Political Economy 2*, 333–344

Kogut, B./A. Metiu (2000), *The Emergence of E-Innovation: Insights from Open Source Software Development*, Reginald H. Jones Center Working Paper, Philadelphia

— / — (2001), Open Source Software Development and Distributed Innovation, in: *Oxford Review of Economic Policy 17*, 248–264

Kollock, P. (1999), The Economics of Online Cooperation: Gifts and Public Goods in Cyberspace, in: M. A. Smith/P. Kollock (eds.), *Communities in Cyberspace*, London, 220–242

— /M. Smith (1996), Managing the Virtual Commons: Cooperation and Conflict in Computer Communities, in; S. Herring (ed.), *Computer-Mediated Communication: Linguistic, Social, and Cross-Cultural Perspectives*, Amsterdam, 109–128

Kreps, D. M. (1997), Intrinsic Motivation and Extrinsic Incentives, in: *American Economic Review 87*, 359–364

Krishnamurthy, S. (2002), Cave or Community? An Empirical Examination of 100 Mature Open Source Projects, http://opensource.mit.edu/papers/krishnamurthy.pdf

Lakhani, K./E. von Hippel (forthcoming), How open source software works: 'Free' user-to-user assistance, in: *Research Policy*

— /B. Wolf/J. Bates/C. DiBona (2002), The Boston Consulting Group Hacker Survey, http://www.osdn.com/bcg/bcghackersurvey.pdf and http://www.osdn.com/bcg/bcghackersurvey-0.73.pdf

Ledyard, J. (1995), Public Goods: A Survey of Experimental Research, in: A. Roth/J. Kagel (eds.), *Handbook of Experimental Economics*, Princeton, 111–194

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

— / — (2002b), *The Scope of Open Source Licensing*, Harvard Business School Working Paper, Boston

Levi, M. (1988), *Of Rule and Revenue*, Berkeley

— (1991), Are There Limits to Rationality?, in: *Achives Européennes de Sociologie 32*, 130–141

Lewicki R. J./Bunker, B. B. (1995), Trust in Relationships: A Model of Trust Development and Decline, in: B. B. Bunker/J. Z. Rubin (eds.), *Conflict, Cooperation, and Justice*, San Francisco, 133–174

Lindenberg, S. (2001), Intrinsic Motivation in a New Light, in: *Kyklos 54*, 317–343

March, J. G. (1991), Exploration and Exploitation in Organizational Learning, in: *Organization Science 2*, 71–87

Markus, M. L./B. Manville / C. E. Agres (2000), What Makes a Virtual Organization Work? in: *Sloan Management Review 42*, 13–26

Merton, R. K. (1993), *On the Shoulders of Giants*, Chicago

Meyerson, D./K. E. Weick/R. M. Kramer (1996), Swift Trust and Temporary Groups, in: R. M. Kramer/T. R. Tyler (eds.), *Trust in Organizations: Frontiers of Theory and Research*, Thousand Oaks, 166–195

Michelman, F. E. (1982), Ethics, Economics and the Law of Property, in: *Nomos 24*, 9

Milgrom, P. R./J. Roberts (1992), *Economics, Organization and Management*, Englewood-Cliffs

Moon, J. Y./L. Sproull (2000), Essence of Distributed Work: The Case of the Linux Kernel, in: *Firstmonday 5*, 11

North, D. C. (1981), *Structure and Change in Economic History*, New York

— (1990), *Institutions, Institutional Change, and Economic Performance*, New York

Organ, D. W./K. Ryan (1995), A Meta-Analytic Review of Attitudinal and Dispositional Predictors of Organizational Citizenship Behavior, in: *Personnel Psychology 48*, 776–801

Orr, S. W. (2001), The Economics of Shame in Work Groups: How Mutual Monitoring can Decrease Cooperation in Teams, in: *Kyklos 54*, 49–66

Osterloh, M./B. S. Frey (2000), Motivation, Knowledge Transfer, and Organizational Firms, in: *Organization Science 11*, 538–550

— /J. Frost/A. Weibel (2002), *Solving Social Dilemmas: The Dynamics of Motivation in the Theory of the Firm*, University of Zürich Working paper, Zürich

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

— (1998), A Behavioral Approach to the Rational-Choice Theory of Collective Action, in: *American Political Science Review 92*, 1–22

— (2000), Crowding out Citizenship, in: *Scandinavian Political Studies 23*, 3–16

Raymond, E. S. (2001), *The Cathedral and the Bazaar*, Sebastopol

Rheingold, H. (1993), *The Virtual Community: Homesteading on the Electronic Frontier*, New York

Rose-Ackerman, S. (1996), Altruism, Nonprofits, and Economic Theory, in: *Journal of Economic Literature 34*, 701–728

— (1998), Bribes and Gifts, in: A. Ben-Ner/L. Putterman (eds.), *Economics, Values, and Organization*, New York, 296–328

— (2001), Trust, Honesty and Corruption: Reflection on the State-Building Process, in: *European Journal of Sociology 42*, 27–71

Sen, A. K. (1974), Choice, Orderings and Morality, in S. Körner (ed.), *Practical Reason: Papers and Discussions*, Oxford, 54–67

Somaya, D. (2003), Strategic Determinants of Decisions not to Settle Patent Litigation, in: *Strategic Management Journal 24*, 17–38

Stallman, R. (1999), The GNU Operating System and the Free Software Movement, in: C. Dibona/S. Ockman/M. Stone (eds.), *Open Sources: Voices from the Open Source Revolution*, Sebastopol, 53–70

Stukas, A. A./M. Snyder/E. G. Clary (1999), The Effects of 'Mandatory Volunteerism' on Intentions to Volunteer, in: *Psychological Science 10*, 59–64

Tönnies, F. (1920), *Gemeinschaft und Gesellschaft – Grundbegriffe einer reinen Soziologie*, Berlin

Torvalds, L. (1998), FM Interview with Linus Torvalds: What Motivates Free Software Developers, in: *Firstmonday 3*, 3

— /D. Diamond, (2001), *Just for Fun: The Story of an Accidental Revolutionary*, New York

Tuomi, I. (2000), Internet, Innovation, and Open Source: Actors in the Network, in: *Firstmonday 6*, 1

Ullman, E. (1997), *Close to the Machine: Technophilia and its Discontents*, New York

Varian, H. R./C. Shapiro (1999), *Information Rules – a Strategic Guide to the Network Economy*, Boston

Von Hippel, E. (1988), *The Sources of Innovation*, New York

— (2001), Innovation by User Communities: Learning from Open Source Software, in: *Sloan Management Review 42*, 82–86

— /G. von Krogh (forthcoming), Open Source Software Development and the Private-Collective Innovation Model: Issues for Organization Science, *Organization Science*

Von Krogh, G./S. Spaeth/K. Lakhani (2003), Community, Joining, and Specialization in Open Source Software Innovation: a Case Study, in: *Research Policy 32*, 1217–1241

Weber, M. (1949), Objectivity in Social Science and Social Policy, in: H. A. Finch/E. A. Shils (eds.), *The Methodology of the Social Science*, New York, 49–112

Wellman, B./M. Gulia (1999), Virtual Communities as Communities, in: M. A. Smith/P. Kollock (eds.), *Communities in Cyberspace*, New York, 167–194