Identifying Business Barriers and Enablers for the Adoption of Open Source Software

Holck, Jesper; Holm Larsen, Michael; Pedersen, Mogens Kuhn

Working Paper

Identifying Business Barriers and Enablers for the Adoption of Open Source Software

By

Jesper Holck
Michael Holm Larsen
Mogens Kühn Pedersen

No. 10-2004

Institut for Informatik

Handelshøjskolen
i København

Howitzvej 60
2000 Frederiksberg

Tlf.: 3815 2400
Fax: 3815 2401
http://www.inf.cbs.dk

Department of Informatics

Copenhagen
Business School

Howitzvej 60
DK-2000 Frederiksberg
Denmark

Tel.: +45 3815 2400
Fax: +45 3815 2401
http://www.inf.cbs.dk

Identifying Business Barriers and Enablers for the Adoption of Open Source Software

Jesper Holck

Michael Holm Larsen

Mogens Kühn Pedersen

Copenhagen Business School, Department of Informatics

Howitzvej 60, 2000 Frederiksberg

E-mail: {jeh,mhl,mk}.inf@cbs.dk

ABSTRACT

The main research interest in Open Source Software (OSS) has been in answering the questions of why individuals and organizations without economic compensation contribute to OSS projects and how these projects are organized. In this paper we instead focus on managerial decisions for acquisition of OSS and discuss potential barriers for widespread use of OSS. Based on existing literature and a small case study, we develop and discuss the hypothesis that a major barrier may be the "customer" organizations' uncertainty and unfamiliarity with the relationships with OSS "vendors". To develop viable models for these relationships is an important challenge, which we will deal with in a research project, of which this paper should be seen as a first step.

1. INTRODUCTION

In developing our understanding of the Open Source "movement" [36, 39, 47], a multi-perspective analysis needs to be undertaken in order to embrace the complexities of this phenomenon. As this research field is fairly young, mature bodies of theories such as economics [39] often set the development directions and research agendas of the research field. In our literature review, research contributions and contributions from practice are both considered relevant, but we find a lack of research addressing Open Source Software (OSS) from a business perspective – not from a product, developer, community, or industry perspective as often seen [44, 62, 64].

Our focus will be the decision-making challenges for managers in organisations when confronted with OSS. Initial decision-making is of interest and highly important because even though the realised costs at this point in time are relatively small, the dispositioned costs, derived from the decisions and dispositional mechanisms [46], will often be substantial.

This paper should be seen as the first step in a larger research project in which we will deal with a series of research questions:

- What barriers and enablers do organizations experience when they consider adoption of OSS?
- If organizations choose to adopt OSS, which organizational measures do they take to acquire and deploy the new software?
- Are these measures different from measures taken when acquiring and deploying commercial software?
- Do these measures vary across different types of organizations?
- Can we identify a set of "best practices" for organizations when they consider, acquire, and deploy OSS?

Presenting the research outline in this paper, we hope to be able to provide early insights from empirical studies later this year.

The paper is organized as follows: After an overview in section two of OSS literature, we will in section three present our research question: why are OSS products not in more widespread use? In section four and five we will outline our understanding of OSS and OSS use, and then in section six discuss possible answers to our research question. As these answers do not seem satisfactory, we will in section seven present the hypothesis that a major barrier for OSS use is the new relationship between the OSS "vendor" and "customer". On basis of a short case study in section eight, we will in section nine discuss our hypothesis and directions for further research.

2. LITERATURE STUDY

Although the research area of OSS is relatively young, documented research contributions are coming at a rapid pace, investigating OSS from different perspectives.

Several contributions give a good introduction to the many aspects of OSS development, including Raymond's influential book [50] and books by Feller and Fitzgerald [17], Wayner [62], and Weber [63]. Many of the most influential OSS advocates present their views on OSS in [13]. A good introduction to the many research questions relating to OSS are given by Feller and Fitzgerald [16] and von Hippel and von Krogh [60, 61].

The one aspect of OSS that seems to have attracted most interest is the question of *why* individuals and organizations may choose to contribute to OSS projects, delivering time, work, and other resources, apparently without economic compensation. Focusing on the individual contributors, Raymond [50] suggested that OSS projects can be described as a "gift economy", an idea also suggested by Bergquist and Ljungberg [5], where one gift (e.g. a source code contribution) must be paid back with other gifts. Ye et al. [65, 66] suggest learning as the primary motivation; Hars and Ou [24] identify both internal (joy of programming, altruism, identification with a community) and external (self-marketing, building human capital, need for software solution) motivational factors, the external factors having most weight. In an analysis of the Apache project [23], Hann et al. found that active contributors received higher wages from the employers, also suggesting economic motives for their "voluntary" work. Several empirical, demographic analysis of individual contributors to OSS have been performed [12], including the EU-financed FLOSS study [22].

Focusing on organizations and using economic theory, several authors have argued that contributing to and participating in OSS projects under some circumstances can be a viable economic activity, including Benkler [4], Lerner and Tirole [38, 39], Johnson [31], Henkel [27], Edwards [15], and Haruvy et al. [25]. von Hippel and von Krogh [60] propose that OSS development under many conditions can provide a benificial compund model between private investments and collective action. Case studies of business models in relation to OSS have been presented by Dahlander [8, 9], and Bonaccorsi and Rossi [7].

Another research question receiving much interest has been *how* OSS projects work; how are they organized, what methods and techniques do they use to produce software? Several case studies have been presented of both well-known OSS projects, including Apache [18, 41], FreeBSD and Mozilla [28, 29], Gnome [33], and less well-known projects [44]. The quality assurance models used in large OSS projects have been described in [28] and [67] A multitude of factors have been identified as having influence on the success of an OSS project, including norms, values, and cognitive trust [57]; professional core contributors, strong hierarchy, and "social movement" organization [26]; peer review [56], coordination [28, 42], and modularity [31, 45].

Scacchi [52] argues from a more technical perspective that OSS development is often faster, better, and cheaper than traditional software engineering. Paulson et al.

[48] find that, compared with commercial projects, OSS projects show more creativity and faster location and removal of bugs; Kogut and Metiu [34] point to higher efficiency due to concurrent design and test, and avoidance of a strong intellectual property regime. Fitzgerald, on the other hand, identifies several weaknesses of the OSS development model [19], including limited interest in mundane tasks (documentation etc.), murky business models, and lack of strategic nous.

On an industry level, Wayner [62] investigates how the Free Software Movement and in particular Linux erodes market shares of established players in the field, and Mustonen [43] identifies circumstances under which OSS can influence the software markets, even when dominated by monopolies.

3. OUR RESEARCH QUESTION

As mentioned above, one puzzling question has been why organizations and highly competent developers voluntarily invest resources in OSS development. In our research, we will look into the complementary question: why are OSS products not in more widespread use in companies and public institutions? Given that the products are available for free, and many are of high quality, this seems surprising from an economical view.

Based on interviews with MIS managers, Dedrick and West [11] found that the most important driver for OSS adoption was cost, both direct savings (cheap software) and indirect savings (no upgrade fees, lower hardware requirements); barriers included compatibility with current technologies and skills, organizational resources and tasks, and the availability of external technological resources. A case story, based on (mostly successful) adoption of OSS have been described by Fitzgerald and Kenny [20, 21]. Several authors have recommended use of OSS in the public sector, including Seiferth [54] (US Department of Defense), Schmitz and Castiaux [53] (public sector administrations across EU), and the Danish Board of Technology [58] (Danish public sector).

Below, we will present and discuss a number of hypotheses regarding the limited adoption of OSS in commercial settings. First, however, we will discuss the notions of OSS and of being an "OSS user".

4. CHARACTERISTICS OF OPEN-SOURCE SOFTWARE

Like many other terms, the exact meaning of "open-source software" is debatable. The definition offered by the Open Source Initiative (OSI) focuses on the software license. In contrast to proprietary software licenses, which mostly deal with restricting users' rights and limiting vendors' liabilities, open-source software licenses, according to OSI, must provide users a number of rights, including

- Anyone is free to distribute and use the software
- The software source code is freely available

However, the license is not the only characteristic of "open-source software" as most people understand it. Another characteristic [27] is:

- Software developed and maintained through the "open-source model," in which many developers contribute code to a common repository

Because of our focus on organizations' adoption of OSS, we will further limit our interest to:

- Software distributed as application programs, excluding e.g. code libraries
- Software maintained and developed by a mature, active organisation, including
  - Technological infrastructure: common software repository, website, mailing lists for users and developers etc.
  - Organizational infrastructure: division of labor, hierarchy, procedures, plans etc.

This definition will cover all major open-source products, including Linux, Apache, MySQL, Mozilla, Eclipse, and Samba.

5. OPEN SOURCE SOFTWARE USERS

Even though we for an OSS product may be able to identify a "vendor" organization that develops, maintains, and distributes the product, and a number of "customer" organizations that use the product, this is clearly not a complete picture of the many possible ways in which organizations can benefit from open-source products. Based on our literature survey, some of the more important seem to be:

- Direct user, using OSS products for the organization's own administration or production processes. An interesting, recent example is the City of Munich's switch to Linux and OpenOffice.
- Service provider, using OSS products to provide services like web hosting, e.g. Typo Systems, Denmark.
- Software reseller or distributor, selling "packaged" OSS, possibly bundled with proprietary software, e.g. Red Hat's Linux distribution.
- Hardware vendor, bundling proprietary hardware products with OSS, e.g. IBM's computers with preinstalled Linux.
- Publisher, selling books that document and describe OSS, e.g. O'Reilly [47].
- Consultant or systems developer, providing customer solutions based on OSS, e.g. the Danish companies Casalogic and Adapt.

In this paper we will focus on the first of these, using OSS for the organization's administrative or production work. We will explore organizational and managerial tools for providing a viable, adequate and robust platform for OSS users, as such a platform must be a necessary precondition for widespread adoption of OSS.

6. POSSIBLE BARRIERS FOR ADOPTION OF OSS

In this section we will identify and discuss three hypotheses for why OSS is not more widely adopted in companies and public institutions.

6.1. Does OSS fit with recommendations for COTS?

When identifying possible barriers and enablers for adoption of OSS it seems reasonable to investigate contributions from the fields of information systems and software engineering. Most research and textbooks in these fields silently assume that an organization's information systems are developed "from scratch", either by the organization itself or by external consultants or vendors, but this assumption does not hold for organizations' adoption of OSS products. The approach of developing information systems based on ready-made software products (COTS, Commercial-Off-The-Shelf Software) is called CBS (Component Based Systems) [see e.g. 30, 59]; especially a research group headed by Barry Boehm has been interested in this area and has developed tools to assist organizations in managing CBS projects.

Based on COCOMO (COnstructive COst MOdel) [6], this group has developed COCOTS (COnstructive COTS model), providing estimates of required resources for CBS development [10, 51]. COCOTS identifies four primary sources of cost/effort associated with use of COTS products (values in parentheses are estimated averages of total effort) [1]:

- Product assessment – vetting and selecting viable components for integration into a larger system (8%)
- Product tailoring and tuning – configuring components for use in the specific context (28%)
- Glue code development – developing code needed to integrate the component into a larger system (46%). Interestingly, this code is assumed to be hard to write because of constraints in the COTS component design, and potential difficulties in obtaining necessary tools and information from the vendor.
- Larger system/application volatility – supposedly increased maintenance effort due to frequent new releases from the COTS vendor (19%)

Based on a large number of US case stories, the research group has collected a large number of "lessons learned" in relation to use of COTS. Some of the more interesting in relation to our present discussion may be [3]:

Avoid modifications. The reason for this is given in one of the cases as "Porting a COTS product generally implies code modifications to the COTS product. Such modifications, if not incorporated into the source by the supplier, means that every release of the source from the vendor has to be modified in accordance with the custom changes, thus losing some of the benefit from using a COTS product."

Unpredictable evolution of COTS products. The evolutionary nature of COTS products has a profound impact on program cost, schedule, and risks: "Decisions about COTS products … must anticipate that product change will be rapid." Of particular interest, when comparing use of commercial COTS and OSS products is the suggestion that "the use of open systems concepts and interface standards and code escrow agreements may help. Also avoid lock-in to a particular vendor product and minimize the amount of customization and glue code."

Many of the results from Boehm et al.'s research point to advantages of OSS, when compared with commercial alternatives:

- It is possible to modify the software and avoid re-doing this with new versions, if you get the modification included in the OSS distribution
- No need for source code escrow. If the customer is not satisfied with the direction or progress from the OSS project, the customer (or eventually another paid consultant) can "take over"
- All interfaces are public, glue code development should be relatively simple

So, rather than reveal substantial barriers for adoption of OSS, Boehm et al.'s work seems to provide arguments in favor of OSS.

6.2. Are OSS products of poor quality?

An obvious explanation for organizations' limited adoption of OSS might be the products' poor quality.

It is not surprising that OSS advocates on one side and commercial software vendors on the other side have debated the quality of OSS products. Proponents have argued that OSS shows fewer errors, is more efficient, and in more conformance with existing standards; but OSS has also been criticized for often having poorer user interfaces and lack of compatiblity and interoperability. Given the large variation in both OSS [35] and proprietary software products, it is of course close to meaningless to make comparisons on such a general level. Research in this area has not identified substantial differences in quality between OSS and commercial products [37, 40, 49, 55], so we do not expect poor quality to be the most important barrier for adoption of OSS products.

6.3. Do OSS products show high TCO or switching costs?

Even though most OSS can be acquired without cost, e.g. freely downloaded from the Internet, this does not imply that the cost of acquiring and using OSS in an organization is negligible. Commercial software vendors, especially Microsoft, have argued that the total cost of ownership (TCO) of OSS may be higher than for commercial software, because of higher switching and maintenance cost, and poor and/or expensive support. Some industry reports, many sponsored by commercial software vendors, point in this direction, but others show quite differing results. We have not found scientific papers providing solid evidence for or against a possible higher TCO of OSS.

Also, recent research shows that real-life decisions regarding IT investments or outsourcing are not made on basis of economical cost-benefit analyses alone [2], but as much or more on qualitative estimates of the technology's value for the organization, and of how risky the market appears to be [14]. So even though a perceived higher TCO may in some circumstances be important, we do not expect this to be the only or decisive argument against adoption of OSS.

7. A NEW RELATIONSHIP BETWEEN VENDOR AND CUSTOMER

As we not in our, admittedly superficial, analysis in the previous section have found convincing support for the three hypotheses regarding barriers for adoption of OSS, we will in this section develop a new hypothesis: that a major barrier is the unfamiliarity and uncertainty in regard to the relationship between OSS "vendor" and "customer".

When an organization buys a proprietary piece of software, the organization pays the vendor a certain amount of money (maybe through a reseller) and receives the software product in return. This implies, of course, that the software vendor has an important, economic interest in attracting users: the more users, the more customers, the more money. This relationship is (ideally) also beneficial to the customer, as it compels the vendor to produce software that fits the customer's needs.

This relationship between "customer" and "vendor" is fundamentally different, when we consider OSS. As the customer does not pay the vendor for the software product, the vendor has no direct interest in keeping the customers satisfied. For the customer, this has (at least) two important consequences. First, the customer may feel this new vendor relation quite unclear: if the product isn't paid for, what "guarantees" do the customer have for the product's quality and the future relation to the vendor? Second, the customer has to find other means than simply paying for the software, if he wants to persuade the vendor to pay attention to his interests.

One solution for the customer may be to choose to buy the OSS product through a commercial company, in this way transferring to a third party the potential problems of cooperating with the OSS vendor. This can remove some of the uncertainties involved in being an OSS user, changing this role to something more like the familiar role of being a software product customer. Another solution for the customer is to engage in the OSS project. The OSS project may not be interested in "pure customers", but it has obvious and objective interests in contributors. We have found several ways in which organizations can participate in and contribute to an OSS project:

- Source code – improvements, corrections in-house (competence building) and offered to the developer community
- Documentation in-house and for the developer or user community
- Error reports, assist in bug finding and removal for the developer community
- Suggestions for improvements to the developer community
- Supply and maintain technical infrastructure for the community (or donate money to do this)
- Participate in management of the community OSS organization
- Responding to requests for help in user communities
- Participate in and support local chapters of developer/user communities

As described in [32], it is also in a commercial setting important for a software vendor to establish good links to customers and there are many ways to establish these links (bulletin boards, customer groups, pre-release demonstrations etc.), but they are of minor importance, compared with the relationship defined by the

software acquisition, and the activities are typically on the initiative of and sponsored by the vendor organization.

In an open source setting, however, user participation in an OSS project is of major importance in the relationship between "customer" and "vendor", and will be on the initiative of and sponsored by the customer organization.

8. CASE DESCRIPTION

Copenhagen Business Academy (CBA) was founded in 1843 and now provides teaching from 13 sites in Copenhagen. In 2002 CBA had a turnover of Danish Kroner 445M (approx. USD 73M), and the result of the fiscal year was DKK 17M. In 2003 the turnover exceeded DKK 500M with 700 employees and 17,000 students enrolled. CBA offers two advanced studies, and more than 50 different types of training and education.

Large parts of CBA's information systems were developed by the Danish Ministry of Education and has been mandatory for the institution: most importantly the student and teacher administration system (EASY-A), and the financial system (EASY-Ø). In addition to these, CBA has two Microsoft Exchange servers, running as post offices for students and employees, a SAS Institute executive information system, and various educational systems.

Six persons are employed in CBA's IT-group, administrating and supporting the information systems; additionally, a number of teachers have support of the educational systems as part of their job. The annual cost of the IT-group is DKK 7M (approx. USD 1.1M), of which typically around DKK 0.1M is allocated to external consulting services, though in some years this amount may be 10 times larger. The prime goal of the IT-group is to provide efficient and effective administration and support of CBA's IT infrastructure. Hence, a lever for obtaining this goal is to reduce the complexity and variety of employed systems and software.

Except for educational purposes (courses in OpenOffice etc.), CBA does not at present employ OSS. According to the IT director, the primary advantage of OSS should be the low acquisition cost, but because of substantial educational discounts from vendors (primarily Microsoft), this advantage is very limited and the perceived lack of support of OSS is a decisive barrier. He observed that

> "If you buy from Red Hat, there is no support, nobody has checked for security holes, no one updates the drivers, nobody does anything – you are left high and dry. This is a precarious situation."

Another barrier comes from the importance of compatibility with the Oracle database management system (DBMS). Early in the requirements specification phase for the EASY systems it was decided to base these on the Oracle DBMS, and as the costs of now changing to another DBMS would be very high, it is of decisive importance for CBA to choose hardware and software compatible with the Oracle DBMS. Oracle should be compatible with Linux, but – according to CBA – support of the Linux platform seems to have a low priority for Oracle, causing upgrades for Linux to be almost one version number behind upgrades for prioritized platforms like Sun Solaris and Microsoft Windows. So, according to the IT director, a switch

to Linux would seriously increase the need for local system "patches" while waiting for new releases of the Oracle DBMS. Furthermore, CBA is worried about compatibility problems between different Oracle versions, Linux flavors and versions, and hardware. Summing up, CBA is highly vendor dependent, and has excluded OSS due to perceived lack of support, worry of compatibility problems, and slow timing of upgrades.

9. DISCUSSION

In larger organizations, decisions regarding procurement of SW components are seldom taken on basis of the qualities of the individual component alone. Typically, top management will have decided upon a common IT policy and enterprise architecture, constituting a strategic framework for future IT investments. If OSS, as is most likely, is not part of this framework, it will not be adopted in any significant scale – not even if certain OSS products are highly competitive when compared with commercial alternatives. Exceptions may be certain "niche" areas, more or less invisible to company management, like software for researchers or for IT-department servers. Only at the time of IT policy revision will a reassessment of the present platform take place, and only if an open IT-architecture is invited by management will OSS find its way to the decision-making arena. If the organization has little competence in OSS, a migration to OSS will be perceived a high risk option with little or no support from the IT department. Only if decisions on IT strategy are taken at a corporate level, above IT operations management, will OSS become subject to serious evaluation.

This situation may be different in smaller organizations without constraining, strategic IT policies; these organizations might be expected to be more likely to experiment with platforms and new software vendors, including OSS. But, as the example of CBA shows, even smaller organizations may be severely constrained in their decisions: legal obligations and large investments may make it unfeasible to change from commercial components to OSS. Also smaller organizations will only have limited resources to develop competencies necessary for acquiring, deploying, and supporting OSS.

Procurement models and their "fit" with vendors' delivery models are essential when organizations formulate their IT policies. Hence, one answer to our research question from section three is that a major barrier for adoption of OSS is the lack of reliable procurement models, which must include technical (appropriation regarding functionality, security, interfaces etc.), legal (appropriation regarding license), and business elements (appropriation regarding vendor, customer support etc.). In the commercial market, satisfactory and well-proven procedures exist for these elements, but this has yet to evolve and mature for OSS.

It is our hypothesis that organizations will only adopt OSS (in any significant scale) if one of two conditions is met:

- OSS is bundled with hardware products, delivered through commercial vendors. This is what we are now seeing with IBM's and HP's distribution of computers with the Linux operating system. In this way, the OSS is not

really acquired by the organization, but rather delivered as an included subcomponent.

- A credible combination of delivery and procurement models for OSS is found.

So, for OSS to obtain a larger "market share" in the coming years, it will be an important challenge for both users and developers of OSS to establish credible and mutually acceptable combinations of OSS delivery and procurement models. We plan to continue our research in this area by collecting a both broader and more detailed view of how organizations make decisions regarding OSS acquisition and deployment, and on basis of this hopefully we will during the next year be able to present viable models and best practices for adoption of OSS in organizations.

10. REFERENCES

1. COCOTS, University of Southern California, 2001.
2. Bannister, F. and Remenyi, D. Acts of faith: instinct, value and IT investment decisions. *Journal of Information Technology*, *15* (3). 231-241.
3. Basili, V.R. and Boehm, B. COTS lessons learned, n.d.
4. Benkler, Y. Coase's Penguin, or, Linux and The Nature of the Firm. *The Yale Law Journal*, *112* (3).
5. Bergquist, M. and Ljungberg, J. The Power of Gifts: Organising Social Relationships in Open Source Communities. *Information Systems Journal*, *11* (4). 305-320.
6. Boehm, B.W., Horowitz, E., Madachy, R., Reifer, D.J., Clark, B.K., Steece, B., Brown, A.W., Chulani, S. and Abts, C. *Software Cost Estimation with COCOMO II*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2000.
7. Bonaccorsi, A. and Rossi, C. Altruistic individuals, selfish firms? The structure of motivation in Open Source software. *First Monday*, *9* (1).
8. Dahlander, L. Appropriating Returns From Open Innovation Processes: A Multiple Case Study of Small Firms in Open Source Software, Department of Industrial Dynamics, School of Technology Management, Chalmers University of Technology, Gothenburg, Sweden, 2004.
9. Dahlander, L. Appropriating the Commons: Firms in Open Source Software, Department of Industrial Dynamics, School of Technology Management, Chalmers University of Technology, Gothenburg, Sweden, 2004.
10. Dean, J., Oberndorf, P., Vigder, M., Abts, C., Erdogmus, H., Maiden, N., Looney, M., Heineman, G. and Guntersdorfer, M. COTS Workshop: Continuing Collaborations for Successful COTS Development. *Software Engineering Notes*, *26* (1). 61-.
11. Dedrick, J., Gurbaxani, V. and Kraemer, K.L. Information Technology and Economic Performance: A Critical Review of the Empirical Evidence. *ACM Computing Surveys*, *35* (1). 1-28.
12. Dempsey, B., Weiss, D., Jones, P. and Greenberg, J. Who Is an Open Source Developer? A Qualitative Profile of a Community of Open Source Linux Developers. *CACM*, *45* (2). 67-72.

13. DiBona, C., Ockman, S. and Stone, M. (eds.). *Open Sources: Voices from the Open Source Revolution*. O'Reilly, 1999.
14. Dos Santos, B.L. Information Technology Investments: Characteristics, Choices, Market Risk and Value. *Information Systems Frontiers*, *5* (3). 289-301.
15. Edwards, K., An Economic Perspective on Software Licenses - Incentives in Open Source Software. in *8th Annual CTI Conference*, (Copenhagen, Denmark, 2003), CTI.
16. Feller, J. and Fitzgerald, B., A framework analysis of the open source software development paradigm. in *Twentyfirst International Conference on Information Systems (ICIS)*, (Brisbane, Queensland, Australia, 2000), Association for Information Systems, 58-69.
17. Feller, J. and Fitzgerald, B. *Understanding Open Source Software Development*. Pearson Education, London, England, 2002.
18. Fielding, R.T. Shared Leadership in the Apache Project. *Communications of the ACM*, *42* (4). 42-43.
19. Fitzgerald, B. The Mysteries of Open Source Software: Black and White and Red All Over?, 2000.
20. Fitzgerald, B. and Kenny, T. Developing an Information Systems Infrastructure with Open Source Software. *IEEE Software*, *21* (1). 50-55.
21. Fitzgerald, B. and Kenny, T., Open Source Software in the Trenches: Lessons from a Large-Scale OSS Implementation. in *International Conference on Software Engineering (ICSE 2003)*, (Portland, Oregon, USA, 2003).
22. Ghosh, R.A., Glott, R., Krieger, B. and Robles, G. Part 4: Survey of Developers. in *FLOSS Final Report*, 2002.
23. Hann, I.-H., Roberts, J., Slaughter, S. and Fielding, R.T., Why do developers contribute to open source projects? First evidence of economic incentives. in *International Conference on Software Engineering (ICSE)*, (Orlando, Florida, USA, 2002).
24. Hars, A. and Ou, S., Working for Free? Motivations of Participating in Open Source Projects. in *34th Hawaii International Conference on System Sciences*, (Hawaii, 2001).
25. Haruvy, E., Prasad, A. and Sethi, S.P. Harvesting Altruism in Open-Source Software Development. *Journal of Optimization Theory amd Applications*, *118* (2). 381-416.
26. Healy, K. and Schussman, A. The ecology of open-source software development, 2003.
27. Henkel, J. Open Source Software from Commercial Firms -Tools, Complements, and Collective Invention, 2003.
28. Holck, J. and Jørgensen, N. Continuous Integration and Quality Assurance: a Case Study of two Open Source Projects. *Australasian Journal of Information Systems* (Special issue 2003/2004).
29. Holck, J. and Jørgensen, N. Do not Check In On Red: Balancing Anarchy with Control in Two Open Source Projects. in Koch, S. ed. *Free/Open Source Software Development*, IDEA Publishing, 2004.

30. Jain, H., Vitharana, P. and Zahedi, F.M. An Assessment Model for Requirements Identification in Component-Based Software Development. *The Data Base for Advances in Information Systems*, *34* (4).
31. Johnson, J.P. Open Source Software: Private Provision of a Public Good. *Journal of Economics & Management Strategy*, *11* (4). 637-662.
32. Keil, M. and Carmel, E. Customer-Developer Links in Software Development. *Communications of the ACM*, *38* (5). 33-44.
33. Koch, S. and Schneider, G. Effort, Cooperation and Coordination in an Open Source Software Project: GNOME. *Information Systems Journal*, *12* (1).
34. Kogut, B. and Metiu, A. Open-Source Software Development and Distributed Innovation. *Oxford Review of Economic Policy*, *17* (2). 248-264.
35. Krishnamurthy, S. Cave or Community?: An Empirical Examination of 100 Mature Open Source Projects. *First Monday*, *7* (6).
36. Lakhani, K.R. and von Hippel, E. How open source software works: "free" user-to-user assistance. *Research Policy*, *32* (6). 923-943.
37. Lawrie, T. and Gacek, C. Issues of dependability in open source software development. *Software Engineering Notes*, *27* (3). 34-37.
38. Lerner, J. and Tirole, J. The Open Source Movement: Key Research Questions. *European Economic Review*, *45*. 819-826.
39. Lerner, J. and Tirole, J. Some Simple Economics of Open Source. *The Journal of Industrial Economics*, *L* (2). 197-234.
40. Mishra, B., Prasad, A. and Raghunathan, S., Quality and Profits Under Open Source versus Closed Source. in *23rd International Conference on Information Systems (ICIS)*, (Barcelona, Spain, 2002), IEEE.
41. Mockus, A., Fielding, R.T. and Herbsleb, J.D. Two Case Studies of Open Source Software Development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, *11* (3). 309-346.
42. Mockus, A. and Herbsleb, J.D., Why not Improve Coordination in Distributed Software Development by Stealing Good Ideas from Open Source? in *24th International Conference on Software Engineering*, (Orlando, Florida, USA, 2002).
43. Mustonen, M. Copyleft - the economics of Linux and other open source software. *Information Economics and Policy*, *15* (1). 99-121.
44. Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K. and Ye, Y., Evolution Patterns of Open-Source Software Systems and Communities. in *Workshop on Principles of Software Evolution (IWPSE)*, (Orlando, Florida, 2002), ACM, 76-85.
45. Narduzzo, A. and Rossi, A. Modularity in Action: GNU/Linux and Free/Open Source Software Development Model Unleashed, 2003.
46. Olesen, J. Concurrent Development in manufacturing - based on dispositional mechanisms *Intitute for Engineering Design*, Technical University of Denmark, 1992.
47. O'Reilly, T. Hardware, Software, and Infoware. in DiBona, C., Ockman, S. and Stone, M. eds. *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates, Sebastopol, California, USA, 1999.

48. Paulson, J.W., Succi, G. and Eberlein, A. An empirical study of open-source and closed-source software products. *IEEE Transactions on Software Engineering*, *30* (4). 246-256.
49. Payne, C. On the Security of Open Source Software. *Information Systems Journal*, *12* (1).
50. Raymond, E.S. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly & Associates, Inc., Sebastopol, California, USA, 2001.
51. Reifer, D.J., Basili, V.R., Boehm, B.W. and Clark, B. Eight Lessons Learned during COTS-Based Systems Maintenance. *IEEE Software*, *20* (5). 94-96.
52. Scacchi, W. When is Free/Open Source Software Development Faster, Better, and Cheaper than Software Engineering? in Koch, S. ed. *Free/Open Source Software Development*, IDEA Publishing, 2003.
53. Schmitz, P.-E. and Castiaux, S. Pooling Open Source Software: An IDA Feasability Study, European Commission, DG Enterprise, 2002.
54. Seiferth, C.J. Open Source and these United States. *Knowledge Technology & Policy*, *12* (3).
55. Stamelos, I., Angelis, L., Oikonomou, A. and Bleris, G.L. Code Quality in Open-Source Software Development. *Information Systems Journal*, *12* (1).
56. Stark, J., Peer reviews as a quality management technique in open-source software development projects. in *European Conference on Software Quality (ECSQ 2002)*, (Helsinki, Finland, 2002), Springer-Verlag.
57. Stewart, K.J. and Gosain, S. Impacts of ideology, trust, and communication on effectiveness in open source software development teams, 2003.
58. Teknologirådet. Open-source software - in e-government, 2002.
59. Torchiano, M. and Morisio, M. Overlooked Aspects of COTS-Based Development. *IEEE Software*, *21* (2). 88-93.
60. von Hippel, E. and von Krogh, G. Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science. *Organization Science*, *14* (2). 209-223.
61. von Krogh, G. Open-Source Software Development. *MIT Sloan Management Review*, *44* (3). 14-18.
62. Wayner, P. *Free for All*. HarperCollins, 2000.
63. Weber, S. *The Success of Open Source*. Harvard University Press, 2004.
64. Wilson, G. Is the Open-Source Community Setting a Bad Example. *IEEE Software*, *16* (1). 23-25.
65. Ye, Y. and Kishida, K., Toward an Understanding of the Motivation of Open Source Software Developers. in *International Conference on Software Engineering (ICSE 2003)*, (Portland, Oregon, USA, 2003).
66. Ye, Y., Kishida, K., Nakakoji, K. and Yamamoto, Y., Creating and Maintaining Sustainable Open Source Software Communities. in *International Symposium on Future Software Technology (ISFST'02)*, (Wuhan, China, 2002), Software Engineers Association.
67. Zhao, L. and Elbaum, S. Quality Assurance Under the Open Source Development Model. *The Journal of Systems and Software*, *66* (1). 65-75.