

Economics of Open Source: A Brief Literature Overview

Alexander Vostroknutov*
Department of Economics
University College London
Gower Street
London, WC1E 6BT
UK
a.vostroknutov@ucl.ac.uk

February 2002

1 Introduction

In the last few years there was a significant increase of interest in open source software development. The organization of work in open source projects is different from usual or *hierarchical* way (Neus, 2001) of writing software inside commercial firms. Open source software can be re-distributed without royalties and fees; it can be modified by *any* person and be re-distributed again; source code should be provided with the software (Weber, 2000).

For the present moment, there exists huge number of open source projects. Hundreds and thousands of developers from all over the world take part in these projects working in non-controllable and non-coordinated fashion without having any compensation for their work. The three most famous and successful open source projects are Apache web server, Linux operating system and Sendmail – e-mail transfer program. Apache is run on approximately 60% of all web-servers in the Internet. The number of Linux users worldwide exceeds 20 million and continues to grow rapidly. Sendmail is

* This essay is prepared for UCL MSc in Economics course “Economics of Innovation” (MB10).

used on the vast majority of e-mail transferring servers (for more details on the development of these projects see (Lerner, Tirole, 2000; Lakhani, von Hippel, 2000)).

It is reasonable to ask a question: why do open source software projects (OSS projects) receive constant attention of economists? The answer is simple: at first sight it seems that all the participants of OSS projects spend their valuable time and brain resources on doing work, which does not give them any payoff at all. This behavior looks paradoxical and absolutely irrational. However OSS movement is getting bigger and has already shown its ability to develop highly complicated software products, which are capable to compete with traditionally made software. This is a major challenge to economists who are used to think in terms of rationality.

Growing literature addresses the question of what are the motives for people to participate in OSS projects. Among possible answers are career concerns and peer recognition (Lerner, Tirole, 2000; Weber, 2000; Dalle, Jullien, 2001b); inducing manufacturer improvements, setting new standards, reciprocity/reputation effects, and low rivalry conditions (Harhoff et al, 2000). Additional analysis of this question is conducted in the following papers: Johnson (2001) and Dalle & Jullien (2001b) construct formal models of rational agents' behavior in OSS projects; the issue of macro-organization of OSS (gift cultures) is covered in (Kelty 2001; Weber, 2000; Edwards, 2001); the effects of public license agreements on incentives to participate in OSS projects are scrutinized in (Weber, 2000; Dalle, Jullien, 2001b).

Another branch of research inspired by startling example of OSS turned to the exploration of *user innovation communities* in general. The following industries were found to have users innovators: production of new sport equipment (Franke, Shah, 2001; Shah, 2000; von Hippel, 2001), production of equipment to produce "copper-interconnect" semiconductors, clinical chemistry analyzers, and computerized library information systems (Harhoff et al, 2000).

Some other questions are of interest as well. It is primarily: analysis of competition between OS and commercial (or proprietary) software products (Johnson, 2001; Dalle, Jullien, 2001a; Khalak, 2000); analysis of OSS community as a "virtual community of practice" (Neus, 2001); and OSS as a complex public good (Bessen, 2001).

This paper is structured in the following way. Possible explanations of OSS development process and related issues are examined in section two. Section three covers the existing research on user innovation communities. Section four gives links to the useful materials.

2 Explaining Open Source

2.1 Why do programmers participate in OSS projects?

Lerner and Tirole (2000) split this question into two parts. First one is: what are the costs of participating in OSS project? Second one: what are the benefits? Working on an OSS project programmer incurs an opportunity cost of his time. He could spend this time executing some other task and have monetary compensation for that. And, what is more important, being hired by a company, research lab or university, OSS participant does not focus on his primary mission.

Benefits are immediate as well as the costs. First, working on an OSS project, programmer can actually *contribute* to carrying out his primary mission. For example, system administrator can improve the performance of the network he is administrating by fixing the bugs in open source software he uses. More serious incentives are *career concerns* and *ego gratification*. Both can be grouped under common term *signaling incentive*. By participating in an OSS project programmer signals his professional abilities to public. Employers can be interested in hiring talented programmer who took part in a popular OSS project. Other programmers will recognize him as a good programmer if he puts lots of efforts to working on the project.

Harhoff, Henkel and von Hippel (2000) propose the following incentives to participate in OSS projects. (1) *Inducing manufacturer improvements*. Programmer may want to freely reveal his innovation to manufacturers in order for them to improve it and to sell on the market. This can be the case since programmer alone is not able to provide highly robust and universal piece of code together with field maintenance and repair programs. He will benefit from buying these innovations on the market. (2) *Reciprocity and reputation effects*. By revealing an innovation programmer creates “generalized

reciprocity”. Other programmers will be inclined to reciprocate. By opening his innovation programmer can get significant reputational gains as the first person, who made this innovation. (3) *Low rivalry conditions*. If competition between programmers is small, programmer will not suffer a lot from revealing his innovations. This can be interpreted not as incentive but as an absence of disincentive.

Programmers who develop open source software have direct incentive to reveal their innovations since they can gain from using high quality commercial products based on their innovations (for example Linux is an open source project and Red Hat Linux is commercial product based on it). Programmers may want their code to be incorporated into the standard version of open source software since, if the coordinating user group approves the code, they gain a reputational advantage. Low competition between programmers makes no incentive to hide the code as well.

2.2 *Open source vs. proprietary software*

Johnson (2001) constructs simple model of open and closed source (i.e. proprietary) software development. He points out advantages and disadvantages of both approaches and says under which conditions open model is preferred to closed one and vice versa. One of the advantages of OSS development is large *talent pool*. Thousands of people can be involved in building open source software. On the other hand, the number of programmers that can be hired by commercial firms is restricted. However, more important advantage of open source is *better knowledge* of what code actually should be written. Participants of OSS projects know what they want and write it. On the other hand commercial firms are not aware of all possible preferences that users can have, so proprietary products tend to be more universal and to satisfy average user’s needs.

However, OSS development has some disadvantages. Open source is an example of provision of a public good, so free riding should decrease the level of innovation in OSS projects to some extent. Another notion is that participants of OSS projects do not write the code they do not want to write (they simply do not need it or writing it can harm their reputation among other programmers). In this situation a great number of such programs as network protocols, drivers or specialized utilities is written but at the same time some projects are not developed at all. For example, open source word processors

and spreadsheets are still cannot compete with commercial ones (like Microsoft Office). For the same reason OSS tends to be incomplete in the sense that it usually lacks such things as user-friendly interface or detailed documentation.

Dalle and Jullien (2001a) consider the *technological competition* between open source and proprietary software. They look at the following differences between open and closed source development. (1) Innovations in open source are more efficient since they are made by users themselves who know what they need (this was already mentioned above). (2) Proprietary software developers have incentives to realize their software only from time to time. Free bug fixing is rare and users should pay for new versions. OSS is regularly delivered to users; bugs are fixed quickly and efficiently. (3) The performance of proprietary technology depends on investments. Investors constantly face a trade-off between investments and profits; so proprietary projects can lack funding. OSS is free, no investments is needed.

Dalle and Jullien conclude that OSS can win the battle with proprietary software but this depends on several conditions. The crucial point is the efficiency of the organization of open source community. Another key to success of open versus closed source is the adoption of proprietary standards by OSS. Proprietary software has an advantage here since for OSS to spread in the population of users it should be compatible with popular proprietary programs and, on the other hand, proprietary programs should not be compatible with OSS. For example, as a consequence of early adoption, Linux experiences great success on the server market. On the contrary, its stagnation on the PC market can be explained by the absence of attention paid by early Linux users to graphical user interface (GUI). As a result, Linux based systems without proper GUI cannot compete with proprietary products with well-developed GUI.

Other articles mentioned in the introduction develop similar concepts of OSS. Some of them pay more attention to the influence of public licenses on OSS development (Weber, 2000; Dalle, Jullien, 2001b). But in general conclusions are the same.

3 User Innovation Communities

3.1 Sport Equipment

Franke and Shah (2001) describe how innovations are made in communities of sport enthusiasts. They study four examples: sailplaning community, canyoning community (new extreme sport, which combines different rock-climbing techniques), boardercross community (type of snowboarding) and handicapped cycling community. Participants of all these communities are users of sport equipment and some of them do innovate in order to make their sport more interesting. Moreover, innovators share their inventions freely with other members of community, who, in their turn, provide innovators with help and advice. It was found that main forces that keep these communities together are reciprocity/reputation and low rivalry effects. When innovator shares his invention with other members of community, they feel obliged to him and provide him with help. Moreover, good innovators have better reputation in the community. It was noticed as well that the number of shared innovations decreases as rivalry becomes more intense. But nonetheless, arbitrarily high levels of rivalry did not bring to naught the number of freely shared inventions.

The main conclusion of the paper is that users are the only source of innovations in the described communities. Manufacturers, who actually produce sport equipment, adopt users' inventions later and make them popular among wider population of sportsmen. Studying of sportsmen's communities is necessary since there are a lot of similarities between innovations in sport equipment, OSS development and other industries.

Shah (2000) and von Hippel (2001) consider communities alike. They add examples from skateboarding and surfing. Conclusions are mainly the same: manufacturers of sport equipment do not innovate themselves but use freely shared inventions of users.

3.2 OSS and gift economies

Kelty (2001) and Edwards (2001) treat OSS movement as a gift economy, i.e. “a form of meta-market with its own currency” (Kelty, 2001). The currency here is reputation. This way of thinking relates OSS movement with *science*, which is found to be gift economy as well. In both communities reputation plays major role in motivating participants.

4 Useful links

Works described in this overview are mainly unpublished working papers. Good source of materials is <http://opensource.mit.edu/>. Another useful link is <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/>. This is a homepage of Eric Steven Raymond, active leader of OSS movement. Raymond (2000a; 2000b) gives brilliant description of OSS community “from the inside”.

5 References

- Bessen, J. (2001), “Open Source Software: Free Provision of Complex Public Goods”, mimeo, 31p.
- Dalle, J.-M., Jullien, N. (2001a), “Open-Source vs. Proprietary Software”, mimeo, 16p.
- Dalle, J.-M., Jullien, N. (2001b), “‘Libre’ Software: Turning Fads into Institutions?”, mimeo, 16p.
- Edwards, K. (2001), “Epistemic Communities, Situated Learning and Open Source Software Development”, working paper, 24p.
- Franke, N., Shah, S. (2001), “How Communities Support Innovative Activities: An Exploration of Assistance and Sharing Among Innovative Users of Sporting Equipment”, Sloan Working Paper #4164.
- Harhoff, D., Henkel, J., von Hippel, E. (2000), “Profiting from Voluntary Information Spillovers: How Users Benefit from Freely Revealing Their Innovations”, MIT Sloan School of Management WP #4125.
- Johnson, J.P. (2001), “Economics of Open Source Software”, mimeo, 27p.

- Khalak, S. (2000), "Economic Model for Impact of Open Source Software", mimeo, 16p.
- Kelty, C.M. (2001), "Free Software/Free Science", First Monday, Vol. 6, No. 12 (December 2001), http://firstmonday.org/issues/issue6_12/kelty/index.html.
- Lakhani, K., von Hippel, E. (2000), "How Open Source Software Works: "Free" User-to-User Assistance", MIT Sloan School of Management Working Paper #4117.
- Lerner, J., Tirole, J. (2000), "The Simple Economics of Open Source", mimeo, 53p.
- Mockus, A., Fielding, R.T., Herbsleb, J. (2000), "A Case Study of Open Source Software Development: The Apache Server", Proceedings of ICSE'2000, 11p.
- Neus, A. (2001), "Managing Information Quality in Virtual Communities of Practice", In: Pierce, E. & Katz-Haas, R. (Eds.) Proceedings of the 6th International Conference on Information Quality at MIT, Boston, MA: Sloan School of Management.
- Raymond, E.S. (2000a), "The Cathedral and the Bazaar", mimeo, <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/>, 38p.
- Raymond, E.S. (2000b), "The Magic Cauldron", mimeo, <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/>, 40p.
- Shah, S. (2000), "Sources and Patterns of Innovation in a Consumer Products Field: Innovations in Sporting Equipment", Sloan Working Paper #4105.
- von Hippel, E. (2001), "Open Source Shows the Way: Innovation by and for Users – No Manufacturer Required!", Forthcoming, Sloan Management Review.
- Weber, S. (2000), "The Political Economy of Open Source Software", BRIE Working Paper 140, E-conomy Project Working Paper 15.