Beyond the Business Model: Incentives for Organizations to Publish Software Source Code

Juho Lindman[1], Juha-Pekka Juutilainen[2], and Matti Rossi[1]

[1] Helsinki School of Economics, Information Systems Science,
P.O. Box 1210, 00101 Helsinki, Finland
{matti.rossi,juho.Lindman}@hse.fi
http://www.hse.fi
[2] Accenture, P.O. Box 1109, 00101 Helsinki, Finland
j.juutilainen@accenture.com
http://www.accenture.com

Abstract. The software stack opened under Open Source Software (OSS) licenses is growing rapidly. Commercial actors have released considerable amounts of previously proprietary source code. These actions beg the question why companies choose a strategy based on giving away software assets? Research on outbound OSS approach has tried to answer this question with the concept of the "OSS business model". When studying the reasons for code release, we have observed that the business model concept is too generic to capture the many incentives organizations have. Conversely, in this paper we investigate empirically what the companies' incentives are by means of an exploratory case study of three organizations in different stages of their code release. Our results indicate that the companies aim to promote standardization, obtain development resources, gain cost savings, improve the quality of software, increase the trustworthiness of software, or steer OSS communities. We conclude that future research on outbound OSS could benefit from focusing on the heterogeneous incentives for code release rather than on revenue models.

1 Introduction

Traditionally OSS is seen as being developed in a distributed setting by a loosely-knit community of heterogeneous developers who contribute to a software project without always being employed or paid by an institution [10]. The development model has resulted in reliable, high quality software products that have a short development cycle and decreased development costs. Many voluntarily started OSS products have outperformed commercial software with similar functionalities. Successful examples include Apache web server, MySQL database, and Linux operating system. Interest towards the OSS phenomenon has grown among companies wanting to replicate these OSS success stories [6]. To this end, organizations have leveraged OSS in their operations, boosted their offering [20], and built their business on new business and revenue models [9]. On the supply side, fundamental changes have occurred in the development process, reward mechanisms, distribution of development work, and revenue models that govern how profit is gained [6]. On the demand side, the buy or build alternatives that are traditionally available to organizations have been supplemented with OSS [6].

C. Boldyreff et al. (Eds.): OSS 2009, IFIP AICT 299, pp. 47–56, 2009.

In addition to using OSS, some companies have released products under OSS licenses or even initiated completely new OSS projects [5]. We have chosen to focus our research effort on understanding this process, coined outbound OSS. Earlier literature on outbound OSS has focused on the revenue stream of the OSS business [19, 9, 12, 17]. While we agree on the importance of a viable company sustaining a guaranteed revenue stream, the heavy emphasis of the earlier literature on the revenue model might have caused some of the other incentives of the organizations' OSS release to be overlooked.

In this paper, we take the viewpoint of the manager making sense of the changing software landscape rather than the viewpoint of the OSS enthusiast. The aim is to gain empirical insight from the company perspective on releasing software to the open domain and thus our research question is: What are the benefits pursued?

2 Background

There has been a paradigm shift concerning software: companies no longer necessarily consider software products as a source of competitive advantage or as the main source of revenue. Conversely, their actions seem to imply that by releasing the source code they gain more than by keeping it secret. Matt Asay, Novell's director of OSS strategy claims that 99.99 % of the products in the world's economy are commoditized [7]. This means that most of the products do not contain anything unequaled. According to Perens, 90% of the software in any business is not differentiating [1, 18]. In most software products, only a small part (5-10%) is differentiating and the remainder is common to the domain. Ultimately every offering that a company delivers to its customers gets commoditized over time [5]. This means that customers are not willing to pay as much for the commodity components and therefore companies should concentrate on creating new and higher value for them [5]. Developing commodity components in-house is not feasible, because they do not provide any additional value. More value is

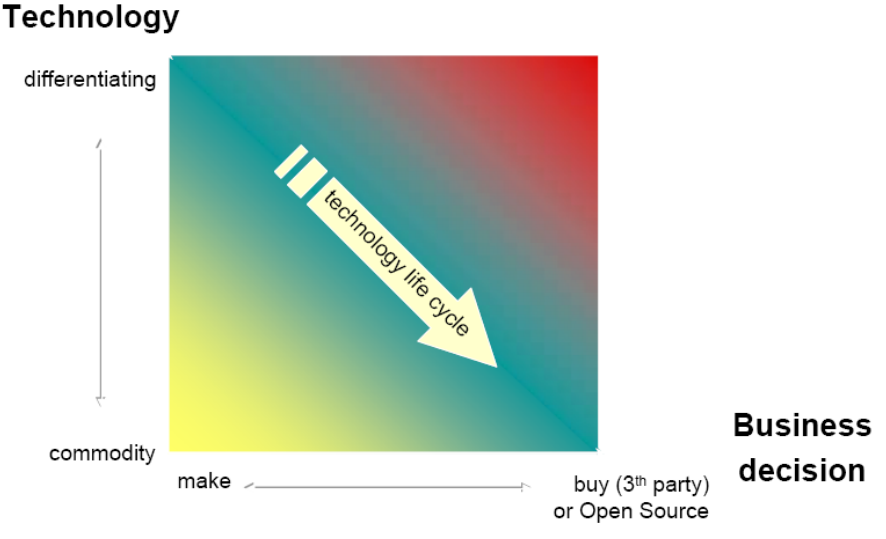

Fig. 1. Commoditization of software (Source: http://www.itea-cosi.org)

created, if companies concentrate on developing differentiating components and acquire commodity components through subcontracting, by using commercial-off-the-shelf products (COTS), or by utilizing OSS.

Outbound OSS approach refers to taking software that is currently sold under a proprietary license and moving it under an OSS license [5]. The opposite process is called inbound OSS, where a company utilizes previously available OSS code and practices inside their own organization [5]. Outbound OSS approach can be characterized as the license-centered approach where a company initiates an OSS project by either releasing the source code of an existing solution to a community as OSS, or initiating an OSS community to develop a new software product [2]. The released source code will then be the basis for the future development of software. West and O'Mahony would call this outbound OSS approach a spinout project because software is first developed internally and later on released to the public under an OSS license [21]. IBM's Eclipse project is one successful example of the outbound OSS approach. After spending more than 40 million dollars on the development of Eclipse, IBM released its source code. By utilizing the outbound OSS approach, there were expectations that IBM could gain development help from other companies, lower the development costs, gain credibility, and gain a better position to compete on the market [23]. Another, not so successful example of source code release would be the Mozilla Netscape browser, where developers needed years of work to make the previously proprietary code feasible after it was published [18].

The outbound OSS approach offers several means through which a company can improve its position on the market. Companies often offer complementary services on top of free software and thus revenue is generated from the sales of the services. A company can pursue cost-reductions and better time-to-market by working collaboratively with the community [5]. The outbound OSS approach can help to reduce development costs if the company succeeds in attracting OSS developers to participate in the development [2, 3]. If the collaboration succeeds, the company can get development resources and be able to improve the product. OSS communities are well-known for having low tolerance for poor contributions, which helps to guarantee good quality [5]. In addition, through frequent releases and with the help of a large community, bugs can be found and fixed quickly [19]. Earlier literature implies that security and reliability can be increased through an OSS-based development because OSS products get tested with the help of a global user community [11]. Finally, by getting involved in OSS projects companies can incorporate OSS ideas into commercial software, spot talented programmers for hiring purposes, and also attract programmers who want to work in an intellectually challenging environment [13].

Outbound OSS approach can also aim for a larger user base and increased feedback. By releasing software as OSS, it is possible to attract new users because the software is free of charge. If there is a commercial counterpart with similar functionality, many users will likely choose the OSS product because it is free. Company can thus gain market share from its competitors and even be able to boost the sales of some related products or services [22]. Thus, the outbound OSS approach can be a powerful method especially if the company has strong competitors [14]. It is also a useful approach in an industry that is dominated by a monopoly [16]. The same reasoning applies to a situation where a company has lagged behind its competitors [5]. Source code release can speed up the diffusion of the product since there are no costs involved in obtaining OSS [2]. Thus, the outbound OSS approach lets companies that could never challenge their competitors on their own, challenge them with the help of an OSS community [15]. The

outbound OSS approach can in particular help small companies with limited resources if they succeed in attracting voluntary developers to help in the software development [3]. The releasing company may gain better competitive position with the help of an active development community. Releasing a low cost alternative also puts pressure on the competitors to lower their prices [2]. Taking part in OSS projects might also arouse interest in the general public and improve corporate image [3].

The outbound OSS approach can help in diffusing new technologies. Approach can be useful if a company has a core infrastructure technology that is an enabler to other products and solutions in the company's portfolio [5]. OSS could then be used as a method to make the company's technology pervasive, or adopted as a standard. OSS development is a useful way to promote standardization [22]. Compatibility is a challenge on the software and hardware markets where there are a vast number of different manufacturers and products. Therefore large companies like IBM want to become active participants in the OSS development and to shape it in their interest [22]. On the other hand, by embracing and supporting OSS projects companies can pre-empt the development of a standard around a technology owned by a powerful rival [13]. Finally, OSS has an effect of encouraging collaboration and it can be used as a way to work with partners and competitors on very large projects, sometimes even involving customer at earlier stages of development [5].

Much of the potential success of outbound OSS will depend on the efforts of people who are willing to work for free [9]. That is why companies need to attract software specialists who are willing to participate in OSS development. However, many voluntary software developers will not participate if they are not treated fairly and provided with freedoms and other intangible "payments" [9]. Thus, in order to succeed in the outbound OSS approach, companies may have to invest considerable amounts of time and money [4].

3 Methodology

Our aim is to show the different benefits companies pursue with the use of an outbound OSS approach. Our selected approach is qualitative and interpretative as we aim to clarify the relevant variables and to understand how companies make decisions about pursuing benefits with outbound OSS [8]. We used three exploratory descriptive case studies and interviews of the company respondents. To be able to formulate a comprehensive view of the outbound OSS approach, in-depth data collection and analysis was needed. In terms of systematic data collection, a series of formal face-to-face semi-structured interviews was conducted. Since the aim was to lay emphasis on the depth, nuance, complexity, and comprehensiveness of the data, interviewing was considered to be the most appropriate method for data collection. Interviews were designed in a way that if a later researcher follows similar procedures when conducting the case study, they should arrive at the same findings [24].

The interviews were conducted as a part of the ITEA-COSI-project. Our selected partners were Philips Medical Systems, Nokia Networks, and European Software Institute (ESI). The selected cases can be seen as typical instances of the phenomenon under study. Five interviews were conducted: three at Philips and one at Nokia and one at ESI. The interviewees were selected so that it would be possible to form a holistic view of the utilization of outbound OSS approach in the case companies. It was desirable that each interviewee would have a comprehensive view of business, close relations to

the OSS community, and a broad understanding of how the OSS approach has impacted on the company. Open questions were chosen to make sure that the answers would be constrained as little as possible. The questions were sent to the interviewees in advance so that they were able to get acquainted with them before the interview. Before and during the actual interview, the interviewees had the possibility to ask for clarifications concerning the questions. During the interview, some of the questions were explained more precisely to guarantee that all the interviewees would understand them in the same way. Some follow-up questions were also posed and clarifications given when necessary. The interviews were conducted in an iterative manner, so it was accepted that responses to certain questions could stimulate new awareness and interest in particular issues, which could then require additional questions to be posed to the interviewee. The estimated time of the interviews was one hour.

The data analysis occurred in three phases. First, the data gathered through the interviews was transcribed. The transcription was conducted by word-for-word basis to guarantee the accuracy of the answers and to avoid misinterpretations. After transcription, all the transcribed interviews were sent to the interviewees so that they were able to read them through and clarify their answers if needed. Only one interviewee clarified some answers. Following this, in a second phase the data was elaborated. The objective was to find relevant information from each case and to develop a rich understanding on the incentives of companies' outbound OSS approach. Finally, in the third phase the results were analyzed and the incentives of the outbound OSS outlined.

4 Cases

4.1 Philips Medical Systems – DVTk

Philips Medical Systems (PMS) manufactures products for the health care industry. Its product portfolio covers for example medical imaging, ultrasound, health care IT, defibrillation, and monitoring modalities. Philips Medical Systems and its partner company created in 2000 a validation application for the medical communication protocol DICOM (Digital Image Communication in Medicine). The application was called DVTk (Dicom Validation Toolkit) and it was distributed within Philips and was also freely downloadable from the Philips Internet pages. After several years of co-development, Philips Medical Systems and its partner company decided to release the DVTk as OSS in June 2005. DVTk is licensed under the LGPL, the source code is available at the SourceForge website and the software is freely available for download.

The DVTk tool itself is free so it does not generate any direct revenues. The long term goal of PMS is that with the help of a user community the quality of DVTk is improved and this will eventually reduce the service and support costs of the tool. The main reason for releasing the source code of the DVTk was to create an independent leading tool for the DICOM validation and service tools. Since the application was earlier closed, the results of validation with DICOM were not always trusted by other organizations. By releasing the application as OSS and by providing the opportunity to review and contribute to the code, trustworthiness of the application was expected to increase. Users could trust the software more because they were able to see that there are no hidden features and see how the product is implemented. In addition, there was an aim to rationalize the software development by releasing the source code of DVTk. Prior to releasing as OSS the distributed development between different sites and between different organizations impacted the efficiency of the work. The development of

the application was running on different isolated source control environments to prevent different developer companies from accessing each other's contributions.

Another reason for opening the code was the intention to create a larger and more active community that could use DVTk, report on bugs, and also help in the development. DVTk application was frequently downloaded even before the code was released, but often the feedback was not very useful. By releasing software as OSS, there was expectation in PMS to have more feedback from the users. In addition, it was expected that PMS could involve more companies in the development of DVTk and this way to reduce development costs.

4.2 Nokia Networks – Benchmark

Nokia Networks is one of the leading telecom equipment providers in the world. It merged in 2007 to form Nokia Siemens Networks. The data was gathered before the merger, so we use the name Nokia Networks when referring to this company. Nokia Networks provides network infrastructure, communications and networks service platforms, as well as professional services to operators and service providers. These solutions include both software and hardware. Nokia Networks uses and integrates OSS products (e.g. Linux) into their products, but software that is ultimately offered to the market is not OSS. Nokia Networks does not currently directly contribute much to OSS projects, but would benefit from some influence on the direction of the development. There have been efforts at Nokia Networks to influence OSS communities by participating in the creation of specifications like OSDL Carrier Grade Linux (CGL) requirements specifications, but the results have not had the desired effect. Our case was aimed to create a benchmarking tool for the selected OSS projects. Earlier Nokia created Network Database Benchmark which is used for measuring the Home Location Register (HLR) type of performance of databases. In our case Nokia Networks was preparing Control Plane Benchmark.

Nokia Networks' goal is that Control Plane Benchmark would highlight possible deficiencies in OSS projects and cause developers to steer projects in the direction Nokia Networks would like them to go. Nokia Networks perceives OSS communities and components as a future-proof solution because commercial companies are getting smaller all the time and their long-term existence is uncertain. The respondent considers OSS communities as a more sustainable option sometimes for software development than commercial companies.

Nokia Networks does not have much official interaction with OSS communities. The communities are often suspicious of big companies and are not especially interested in the products that Nokia Networks provides. Thus communication with OSS communities is mainly through individuals who work in Nokia Networks and are also part of an OSS community. However, these people are not representing Nokia Networks when they are involved in the communities. Nokia Networks has some projects and initiatives to form a closer relationship with OSS communities, for example, a portal to manage its OSS projects and to promote Nokia Networks' involvement in OSS projects. Nokia hosts, contributes to, and sponsors multiple OSS projects. Nokia is, for instance, a strategic developer in the Eclipse Foundation. Nokia Networks is also one of the 20 companies that support Open Source Development Lab (OSDL). With the other members in OSDL, Nokia has developed a kind of future roadmap for Linux distributors. Nokia Networks' aim is to create vision and guidance to enhance Linux and to meet the needs of both the data center and carrier grade market segments.

4.3 European Software Institute – V-Manage

European Software Institute (ESI) was launched as an initiative of the European Commission, with the support of the Basque Government and European companies working in the field of information technology. ESI's main activity is based on helping the software industry to produce software of a higher quality, on time, and at a lower cost. ESI offers consultancy and training services as well as technological support. One of the services that ESI offers to organizations is consultancy for implementing a software product line. The purpose of this consultancy service is to achieve a high level of reuse in all products. ESI provides organizations a disciplined methodology and a suite of tools, called V-Manage, for developing software for embedded systems. Now ESI is planning to utilize the outbound OSS approach and to release the source code of V-Manage. V-Manage helps organizations to develop software especially for software product lines and it is mainly offered to small and medium sized companies.

ESI's service consists of a software called V-Manage and a consultancy service. At the moment, the main source of revenue for ESI is the consultancy service consisting of training, support, and maintenance. Currently, V-Manage is proprietary software licensed to the customers of the consultancy service, but ESI is investigating whether they should license it with an OSS license. In the future the revenues will be generated through the sales of consultancy services. There is an expectation in ESI that opening the code would increase other companies' interest towards the application and eventually increase revenues through the sales of consultancy services. However, it is not expected that obtaining development resources from external parties would result in lower costs. Instead, extra development resources are seen as a way to boost the popularity of V-Manage.

ESI has the aim of providing extension points to V-Manage so that external developers can extend the tool by means of plug-ins. This enables customers and possibly a development community to customize the application according to their own needs and add new features. ESI is planning to release the source code of the extension points and plug-ins and keep the platform proprietary. This way ESI could retain core parts of the V-Manage as closed. The source code of plug-ins would be released under a license that assures that all the modifications and derivative works are distributed and made available under the same license. Initially ESI is planning to use LGPL. By means of this new approach, ESI aims to get software development resources from external partners who are willing to develop the application through extension points. The releasing of the source code could result in an active development community. However, the amount of potential development help is still rather uncertain because the application is very specific so it is not likely to attract a large number of developers. Because of the special nature of the tool, it is expected that developers will more likely be companies than individuals.

5 Incentives for Openness

Probably the best known classification of different OSS revenue models is the one presented by Hecker [9]**.** Hecker's revenue models concentrate mainly on the cash flow between the company and its customers. However, our empirical findings demonstrate that companies also have incentives other than revenue for utilizing the outbound OSS approach. Actually, the only case in our data which can be categorized

according to Hecker's classification is ESI's V-Manage. ESI's approach is consistent with Hecker's support seller model where revenues are generated from selling associated services. By means of the outbound OSS approach, ESI aims to increase the popularity of V-Manage and to boost its revenues through the sales of consultancy services. However, the source code of V-Manage is currently not opened and likely will not be opened at all.

It was evident that the case companies perceive the commercial potential of the outbound OSS approach. Companies have various incentives for releasing the source code of their software. These different objectives also have influence on how outbound OSS is applied in practice. Outbound OSS approach is considered to be suitable for companies whose main business is not the software itself. This implies that a company does not necessarily risk its business by releasing the source code. Instead, revenues are generated for example through the sales of different services. Below are the different incentives categorized in a table format (Table1).

Table 1. Incentives per case company

PMS	Nokia Networks	ESI
	Steer OSS community	Steer OSS community
Obtain development resources		Obtain development resources
Gain cost-savings		
Improve the quality of SW		Improve the quality of SW
Increase trustworthiness of SW		
Promote standardization		

ESI's strategy seems to be that by opening parts of V-Manage companies may become more interested in the tool because they are able to customize it to their own needs and ultimately ESI would generate revenue by consultancy services. Instead, the objectives of neither Nokia Networks nor Philips Medical Systems are directly related to generating revenues through OSS. PMS' goal is to rationalize the software development, create a de-facto standard, and to try to form an active development community. Through the outbound OSS approach, PMS aimed to gain external development resources and improve DVTk. The PMS respondent also maintained that OSS can increase the trustworthiness of DVTk because everyone is able to see how it is implemented. Nokia Networks' objectives notably differ from the goals of PMS and ESI. Nokia Networks tries neither to generate revenues nor gain development resources through the outbound OSS approach. Nokia Networks is developing benchmarking tool to be used by OSS communities. This tool is then released as OSS. The aim of Nokia Networks is it could then leverage the OSS communities through these tools.

It seems that the case companies have very different objectives when they chose the outbound OSS approach. It seems that ESI is the only company having a revenue incentive to release the source code. However, it is evident that financial reasons play a role also with Philips Medical Systems and Nokia Networks. In PMS it is considered that the DVTk project may have an indirect impact on total revenues of PMS. PMS's goal is that by improving the DVTk the service and support costs will decrease. Nokia Networks aims to gain cost savings if they succeed in steering OSS communities because the company will get software products that are implemented according to Nokia Networks' needs.

6 Conclusions and Implications

The objective of this paper was to investigate incentives for commercial companies to release software source code. Revenue models were not the primary concern for any of the case companies. The role of revenue models was considered, but the decisions were not incentivized by direct revenue streams.

Although commercial actors are coming into terms with releasing source code they need to tackle practical concerns. One of the main problems was that companies' OSS products are specialized to niche markets that fail to attract a large population of developers. Another challenge is that companies were willing to utilize OSS resources, but they do not always have plans to compensate for the acquired benefits. The outbound OSS approach also highlights some challenges that a company can confront after the source code is released. Based on our analysis, it seems that these challenges are mainly related to collaboration with OSS communities and maintenance of the code base. Voluntary OSS developers will only participate in software development if they find the project interesting. Thus, gaining contributions from the OSS community is not certain. If the software is very specialized and does not interest the general public, the company might confront difficulties in attracting developers. The company also has to be aware that the community's objectives and timetable in software development will most likely differ from the company's own goals. In order to succeed, the company should create a strategy on how it is going to attract developers, motivate them to participate, and steer them so that the company's objectives will be reached.

It should also be noted that the cases in the paper are at very different stages of their OSS activities, and as such are unlikely to give direct applicable solutions to other companies. They do serve as empirical account of what the incentives for commercial companies are, and hopefully help to refocus research beyond revenue models to the multitude of different company incentives.

Acknowledgements

The authors thank the ITEA-COSI project.

References

1. ITEA-COSI-project, `http://www.itea-cosi.org/` (accessed 14.11.2008)
2. AlMarzoug, M., Zheng, L., Rong, G., Grover, V.: Open Source: Concepts, Benefits, and Challenges. Communications of the Association for Information Systems 16(37), 756–784 (2005)
3. Bonaccorsi, A., Rossi, C.: Comparing Motivations of Individual Programmers and Firms to Take Part in the Open Source Movement: From Community to Business. Knowledge, technology and policy 18(4), 40–64 (2006)
4. Dahlander, L., Magnusson, M.: Relationships between open source software companies and communities: Observations from Nordic firms. Research Policy 34(4), 481–493 (2005)
5. Fink, M.: Business and Economics of Linux and Open Source. Prentice Hall, New Jersey (2002)
6. Fitzgerald, N.: The Transformation of Open Source Software. MIS Quarterly 30(3), 587–598 (2006)

7. Goth, G.: Open Source Business Models: Ready for Prime Time. IEEE Software, 98–100 (November/December 2005)
8. Gray, D.E.: Doing Research in The Real World. Sage Publications, California (2004)
9. Hecker, F.: Setting Up Shop: The Business of Open-Source Software. IEEE Software 16(1), 45–51 (1999)
10. Hertel, G., Niedner, S., Herrmann, S.: Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. Research Policy 32(7), 1159–1177 (2003)
11. Krishnamurthy, S.: A managerial overview of open source software. Business Horizons 46(5), 47–56 (2003)
12. de Laat, P.B.: Copyright or copyleft? An analysis of property regimes for software development. Research Policy 34, 1511–1532 (2005)
13. Lerner, J., Tirole, J.: The Open Source movement: key research questions. European Economic Review 45(4-6), 819–826 (2001)
14. Lerner, J., Tirole, J.: Some Simple Economics of Open Source. Journal of Industial Economics 50(2), 197–234 (2002)
15. Markus, M.L., Manville, B., Agres, C.E.: What Makes a Virtual Organisation Work – Lessons From the Open Source World? Sloan Management Review 42(1), 13–26 (2000)
16. O'Mahony, S.C.: Dissertation: The emergence of a new commercial actor: community managed software project (2002), http://opensource.mit.edu/papers/omahony.pdf (accessed 14.11.2008)
17. Osterwalder, A., Pigneur, Y., Tucci, C.: Clarifying business models: Origins, present, and future of the concept. Communications of the Association for Information Systems 16, 1–25 (2005)
18. Perens, B.: The emerging economic paradigm of Open Source. First Monday 10 (special issue 2: Open source) (2005)
19. Raymond, E.S.: The Cathedral and the Bazaar (2000), http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/(accessed 14.11.2008)
20. Rajala, R., Nissilä, J., Westerlund, M.: Revenue Models in the Open Source Software Business. In: St. Amant, K., Still, B. (eds.) Handbook of research on open source software – Technological, Economic, and Social Perspectives, New York, Hershey, pp. 541–554 (2007)
21. West, J., O'Mahony, S.: Contrasting Community Building in Sponsored and Community Founded Open Source Projects. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences, Waikoloa, Hawaii, p. 196c (2005)
22. Wichmann, T.: Firms' Open Source Activities: Motivations and Policy Implications. Free/Libre Open Source Software: Survey and Study, FLOSS Final Report, Berlecon Research GmbH (2002), http://www.berlecon.de/studien/downloads/200207FLOSS_Activities.pdf (accessed 14.11.2008)
23. Woods, D., Guliani, G.: Open Source for the Enterprise: Managing risks, reaping rewards. O'Reilly Media, Sebastopol (2005)
24. Yin, R.K.: Case Study Research: Design and Methods, 2nd edn. Sage Publications, California (1994)

Mentioned OSS Projects

DVTk http://www.dvtk.org/

Network Database Benchmark http://hoslab.cs.helsinki.fi/homepages/ndbbenchmark/