The Economics of Open Source Software Development
Jürgen Bitzer and Philipp J. H. Schröder (Editors)

1

The Economics of Open Source Software Development: An Introduction

Jürgen Bitzer and Philipp J. H. Schröder

ABSTRACT

This chapter introduces the fundamentals of Open Source Software, its nature, the central economic aspects and the key mechanisms of its development. Furthermore, we present the themes of the book and provide a first overview for the reader by giving short summaries of its chapters.

Keywords: Open Source software, software industry, open source development, innovation.

JEL Classification: H41, L31.

1.1 INTRODUCTION

At first glance, Open Source Software (OSS) development appears to present a stark contrast to traditional production and innovation methods and an unlikely contestant in the marketplace.[1] Some of the most striking features are that OSS development is based on the contributions of volunteer programmers, that these programmers only associated with each other through informal communities, that

[1] The term 'open source' refers to the fact that the program source code is – in contrast to the source code of proprietary software, which is only distributed in compiled machine code – accessible, available and thus alterable by its user.

the resulting software products are made available for free, and that this unconventional development method is able to produce software of high complexity and extraordinary quality (e.g., Raymond 2000a,b, Feller and Fitzgerald 2002).

To grasp the success and size of the phenomenon, consider the following prominent examples: the OSS operating system Linux accounts for a 38% share of the server operating system market (Bitzer 2004) and is supported by major hardware firms such as SUN, Compaq/HP, IBM and Siemens. The OSS Web server 'Apache' captured a market share of 70% by December 2005.[2] The open source Web browser 'Firefox' has been able to snatch a 5% market share from Microsoft's Internet Explorer over the six-month period from May 2004 to October 2004 (Festa 2004). Moreover, the spectrum of applications available as OSS spans an impressive range from home automation to space missions.[3]

Obviously, the emergence of such software has fundamentally changed the software business. The simple facts that production is carried out by voluntary private programmers and that the marketable outcomes are supplied to consumers for free have resulted in far-reaching effects on market structures, business models and innovation processes in the software industry. While incumbents had to adapt their strategies to this newly emerging competition, OSS-based firms had to develop viable business models enabling them to generate profits. These fundamental changes in the software industry have subsequently attracted the interest of analysts, businesspeople and researchers in economics and management science.

This book attempts to capture the most important lines of research dealing with the economics of OSS and its development. Research on these issues has made substantial progress in recent years, both in identifying particularities and in offering explanations for them. The aim of this book is to provide the reader with an accessible and relevant insight into this rapidly evolving literature. We have aimed at focusing on new and promising directions in the OSS research field, attempting to provide the reader with a reasonably broad yet concise and accessible selection of some of the most novel and fascinating research on the topic. At the same time, we try to avoid the risk of simply replicating insights that have already found their way into scholarly journals.

Apart from providing the first major literature review on OSS of its kind (Chapter 2), the present book identifies and is structured around three broad themes. First we have included a number of chapters dealing with the interaction of for-profit firms with OSS (Chapters 3, 4 and 5). These chapters address issues ranging from the motivation of for-profit firms and their relation to OSS communities, to the identification of sustainable OSS-based business models. A second

[2] Cf. information of [http://news.netcraft.com/archives/web_server_survey.html].

[3] Linux-based clusters are applied in an increasing share of simulation and control tasks by NASA and ESA. Furthermore, OSS was used for several tasks in NASA's 2003 'Mars Exploration Rover' mission (Barcomb 2004). The websites of NASA and ESA provide further examples of applications of OSS by these institutions.

important trend presented in this book is the issue of programmer shortage, i.e. the problems that a growing number of OSS projects may exhaust the available pool of voluntary programming capacity, and how OSS communities attempt to tackle this issue (Chapters 6, 7 and 8). Thirdly, the book deals with the issue of OSS innovation by addressing the organizational structures within the OSS development process, including a look at the internal community structure as captured via social network analysis, and by considering the impact of OSS on innovation in the software industry (Chapters 9, 10, 11 and 12).

The remainder of the chapter is structured as follows. In the next section we provide a primer on OSS and the surrounding development process including a sketch of how OSS-related activities actually generate revenues. Section 3 presents short summaries of each of the chapters included in this book. Section 4 concludes the chapter.

1.2 THE CENTRAL FEATURES OF OSS DEVELOPMENT

A number of common features have evolved in various OSS communities and projects, defining our understanding of what OSS development is. The dominant and recurring theme – probably because it creates the starkest contrast to the standard concept of *homo economicus* – is the volunteer programmer/worker, paired with the closely related concept of the user programmer.

Open Source Software is mainly programmed by *volunteers* who engage in such projects free of charge. Obviously, a product which is developed by volunteers and subsequently given away for free is unusual on both the input and the output side of the development process, yet the issues of interest on the input side are on the one hand the 'motivation of participants' and on the other hand 'community management'. As volunteer programmers dedicate their time and effort free of charge, even though they are not able to exclude others from the benefits of their work, the motivation of people participating in an OSS project are obviously different from those hired to carry out programming work in a commercial enterprise (e.g., Torvalds and Diamond 2001, Hertel et al. 2003, Bitzer et al. 2004, Lakhani and Wolf 2005). Understanding the motivation of volunteer participants in OSS projects is therefore crucial for understanding the development process. However, it must be noted that a growing number of firms pay employees to program OSS (e.g., IBM, which invests in developing and adapting OSS to their hardware and software), yet, when looking at the OSS phenomenon as a whole, these 'paid' OSS developers are still in the minority. Programmer communities expand and collapse, displaying high gross flows of entry and exit (e.g., von Krogh et al. 2003). While they are only loosely associated through newsgroups, 'tacit' knowledge of the project is passed on as explicit knowledge through project websites, and, more importantly, it is embedded in the source code of the software itself. Individual OSS projects can

attract substantial numbers of programmers. One of the biggest OSS projects is Linux, with its estimated 10,000 active developers. Furthermore, the SourceForge.net repository of OSS projects, on its own, hosts 106,628 OSS projects with 1,184,551 registered contributors.[4]

The programmers of OSS are often *user programmers*, meaning that they engage in a project because they are not satisfied with the existing software or simply because the required software feature does not exist. Most famously, Linus Torvalds needed a Unix for his PC, resulting in Linux (Torvalds and Diamond 2001), Eric Allman needed a more efficient email server, resulting in 'Sendmail', Larry Wall needed a tool to automatically generate Web pages resulting in 'Perl', and finally Don Knuth needed a convenient tool for typesetting documents resulting in 'TEX'. Thus OSS programmers often benefit directly from developing and improving the software in question.

Another striking feature is that OSS is in principle *available for free*. The central feature of OSS is its freely available source code – in contrast to that of proprietary software – which is accessible and therefore alterable by its users. To ensure that OSS remains 'open', several software licences have been developed. Some licences even enforce that any additions made by one programmer/user must be made available at no cost to all other programmers/users as well. The most well-known of these licences is the GNU General Public Licence.[5] In fact, the cornerstone of the OSS movement is its ability to provide coherent and incentive-compatible licensing schemes. The basic idea and main aim of open source is free usage and the possibility for further development by the user. In reality, the terms 'free' and 'open source software' cover a number of varying forms of software licensing schemes, but all have in common the goal of ensuring free access for the user and preventing the obvious risk of a commercial 'hijacking' of the software. The resulting feature of availability or, more correctly, non-excludability paired with the non-rivalry in consumption – due to the fact that software is an immaterial good, i.e. the use of the program code by one individual does not affect the use of it by another individual – implies that OSS fulfills the central characteristics of a public good. Accordingly it is often understood and modelled as such (Johnson 2002, Bitzer and Schröder 2005, Chapter 3 of this book). In contrast, proprietary software induces excludability via copyright licences and the distribution of compiled software, etc. *Commercial software* prohibits or limits usage, modification and duplication through its licensing conditions or simply the unavailability of the source code.

---

[4] Data from December 2005.

[5] For details on the GNU GPL see the website of the Free Software Foundation (FSF) www.fsf.org. The terms *free software* (introduced by the FSF) and *open source software* (introduced in 1998 by a group of programmers in the open source meeting in Palo Alto, California) are nearly interchangeable. Arguably, the term *open source software* holds the danger that the aspect of the free software concept, namely the free distribution and modification of the software, is not sufficiently emphasized.

An important feature of the OSS phenomenon concerns the organization of the production process. Due to the involvement of volunteers, it is obvious that the development process has to be organized differently than in commercial software enterprises. Obviously, volunteers cannot be forced to carry out specific tasks, and this has to be taken into account in creating incentives and designing decision-making processes – indeed managerial structures – within OSS projects. In fact, most of the major OSS projects effectively operate with several layers of involvement: active users, core developers and project leaders, each with different functions and responsibilities. Identifying the incentive structures and decision-making processes on the one hand and analysing their emergence and development on the other are of considerable importance for the research, as this might point to new organizational models in knowledge industries that enable the integration of volunteers. In particular, the fact that the volunteer programmers organize themselves into fairly loose *community structures* has attracted substantial attention (e.g., Krishnamurthy 2002, Franke and Shah 2003). Such communities, which form for individual projects, have several important features. In most instances these communities are open, in principle allowing any outsider access. Furthermore, within each ongoing OSS project, the names of the 'patrons' are distributed with the source code of their piece of software, which includes a list (e.g., the update log) of all contributors to the project. In this way, programmers receive increased recognition and status within the community, and their climb through the different functions or layers of the project community is determined by the volume and quality of their contributions. Furthermore, the feature of such update logs has promoted the interpretation that participation in such programming communities may simply serve as job signals (e.g., Lerner and Tirole 2002). But other factors such as 'community identification' (Hars and Ou 2002) matter as well. Programmers view the open source community as a family and are therefore willing to do things for the good of the whole, even when they do not benefit personally. Of course, 'membership' in the OSS community entails the obligation for the individual programmer to follow internal rules, i.e. to publish the source code of his software. What is perhaps most surprising is that these communities are scattered across the globe. Geographic proximity of programmers appears to play little or no role in determining the success of a project.

The *flow of information* within and around OSS projects is extremely dense and open to everyone. The Internet is the dominant tool facilitating this information flow. This close-to-costless information is a crucial characteristic of the open source development process. Information about new and ongoing projects is compiled on websites and in newsgroups. Bugs, bug fixes, feature demands etc. are all listed and made available to programmers inside and outside the community. Furthermore, there is a huge informational content in source code itself. The source code enables other programmers both to learn specific programming steps, and to understand, criticize and improve existing programming solutions.

In this respect, software is a unique product because the source code is both the construction blueprint and the product itself. Being able to access and read source code facilitates – intentionally or not – a collective understanding of the development process that can hardly be replicated in other products.[6]

Despite the particularities on the input and the output sides of the OSS development process, it frequently results in marketable software products which ultimately compete with commercial software. Thus the emergence of the public good OSS has far-reaching effects on the software business. The most obvious is the effect of creating competition. Due to its 'costless' development, several OSS products have successfully entered formerly highly concentrated markets, overcoming entry barriers based on the cost advantages of incumbents. Another important effect of the emergence of OSS products is the search for new business models enabling firms to generate revenues based on OSS, and thus point commercial enterprises to alternatives to their traditional software development model. Even though the source code and thus the software may be available for free, OSS has also considerable earning and profit potential. But how does one make money with OSS? Thus far, the literature has largely overlooked the interaction between *for-profit firms and OSS*. In the present book we have included several chapters that provide new insight into the extent and nature of this interaction, and how the firm–OSS relation may take shape and evolve in the future. A visible effect of the relation between for-profit firms and OSS is that an ever-increasing share of OSS development is conducted within firms, i.e. either firms that base their entire business model on OSS or firms where employees, with the full knowledge of their superiors, devote substantial time and effort towards the improvement and development of OSS projects. Notable examples of the latter are IBM and Novell.

The most important way of earning money with OSS is by offering software-related services. The types of services range from compilations of OSS to customization and maintenance of OSS. Well-known examples of compilations of LINUX and related software – the so called 'distributions' – are SuSE, Red-Hat and Mandrake, Debian GNU. What this entails is that even though the source code of LINUX is freely available, and thus can be installed and run by anyone on their computer, substantial technical expertise is needed to actually compile a working version from the source code. Thus, distributions offer an automated installation of Linux and connected OSS even for inexperienced users. The distributions arrive with an installation DVD and handbook, quite similar to commercial software packages.

Another important commercial aspect of OSS is the customization and maintenance services offered. The openness of the source code allows experts to develop

[6] Indeed, commercial software development firms frequently overlook this opportunity, and instead – aiming at a more directed development process – keep developer teams separated, and programming towards pre-announced module interfaces without access to each other's source code.

detailed customizations of the OSS in question. Furthermore, OSS programmer communities do not offer any 24-hour support for the software developed. However, the availability of professional support is the prerequisite for using OSS in critical business applications. A whole layer of firms has evolved to meet these particular customer needs regarding OSS, examples of which range from small local enterprises to big firms like IBM. Thus, by generating revenues from OSS, commercial enterprises pave the way for using OSS in commercial enterprises.

1.3 THE CHAPTERS OF THIS BOOK

Research on OSS has made substantial progress in recent years, both in identifying particularities of the phenomenon and in offering economic explanations for them. Social science research on the topic is growing at an impressive rate and has created a rich and diverse body of literature. Significantly lacking was a comprehensive overview of the state of research pointing out future research areas. Maria Alessandra Rossi fills this gap in Chapter 2 'Decoding the Free/Open Source Software Puzzle: A Survey of Theoretical and Empirical Contributions' of this book. Her already widely cited paper is the first of its kind and has structured the available literature into several broad themes. While the initially dominating question in the economics literature was the puzzle of the OSS phenomenon – namely, why people contribute to a freely available public good – her survey omits and skips the earlier discussions and proceeds directly to the current debate on the complexity and heterogeneity of the OSS phenomenon. First, the chapter reviews the main aspects of the institutional OSS environment, starting out with the available theoretical and empirical evidence on the motivation of contributors. Second, the chapter addresses the literature dealing with the governance structures of OSS development and community organization. Third, it deals with the interaction of OSS with commercial firms. Fourth, it covers the contributions to the literature dealing with intellectual property rights, licensing arrangements and other legal issues. And fifth, it reviews research on public policy with respect to OSS, e.g. the pros and cons of government support. The recurring theme in all these branches of the literature is that OSS is a highly heterogeneous phenomenon, indeed more than one isolated phenomenon. In this literature review, Rossi also succeeds in identifying new and important research questions surrounding OSS.

In Chapter 3, 'Open Source Software: Free Provision of Complex Public Goods', James Bessen considers the role of for-profit firms in the OSS development process by modelling a 'private provision of a public good' framework that includes firms. Starting from the observation that although individual volunteer contributions may initially have been the driving force behind the OSS phenomenon, recent years have seen the increasing involvement of firms and their employees. Many large firms such as IBM, HP, Computer Associates and

Novell have channelled substantial resources into OSS development. The chapter highlights product complexity and disparate customer needs, which rationalize that private agents/firms invest sufficient effort in the development of a public good, without having any property rights. While traditional commercial software pre-packaging limits the range of product features available to customers, and application program interfaces can only partly imitate a full range of features, an OSS product can – thanks to its flexibility – meet highly complex customer needs. Thus, one of the driving forces behind the existence of the privately provided public good OSS is that it extends the range of products available to consumers, complementing, rather than replacing, proprietary software provision. This model makes it possible to answer the question of why profit-seeking firms, and not just individuals, may find it worthwhile to contribute to OSS development.

Cristina Rossi and Andrea Bonaccorsi present a related perspective on for-profit firm involvement in OSS development in Chapter 4, 'Intrinsic motivations and profit-oriented firms in Open Source software. Do firms practise what they preach?'. This chapter complements the work of James Bessen presented in Chapter 3, by providing ample empirical evidence on the incentives of firms that do business with OSS. The data that they present is based on a survey of 146 Italian OSS-based firms. While individual programmer motives have received a great deal of theoretical and empirical attention, the motives of firms have been under-researched, and this chapter is the first to systematically gather data on the incentives of firms that supply Open Source–based products and services to their customers. Rossi and Bonaccorsi establish that – contrary to what one might expect – intrinsic, community-based motivations that the literature thus far has attributed to individual programmers also matter for firms and businesses involved in OSS. Yet they also find a range of firms that display a stark contrast between the proclaimed positive attitudes towards OSS and the actual firm behaviour. In documenting these conflicts, the chapter provides important new insights concerning the underlying entrepreneurial decisions that lead to the establishment of Open Source–based businesses.

Next, in Chapter 5, 'Business models and community relationships of open source software firms', Linus Dahlander and Mats G. Magnusson explore further the theme of for-profit firms' involvement in OSS development. In particular, their work deals with the relationships that OSS-based firms maintain with OSS communities, and examines how they evolve over time and what they have to do with the firms' overall business strategies. They find that an important factor determining an OSS-based firm's ability to create a sustainable business model is its capability to forge a good relationship to the OSS community, which can give it an advantage over competitors. It turns out that in order to harvest the potential benefits of the communities, firms need to develop and maintain substantial in-house capabilities in order to assimilate and align their own work with that of the OSS community. Their chapter shows that heavy involvement in the activities of

programmer communities does, on the one hand, open up access to the resources of the OSS community but, on the other, it limits strategic options. Thus it is crucial that the gap between the firms' interests and values and the communities' interests and values must be kept at a minimum. Dahlander and Magnusson analyze this interaction in terms of the social capital and relational capabilities of the firm, and in doing so provide a novel perspective on the business strategies deployed by OSS-based for-profit firms.

In Chapter 6, Martin Michlmayr and Anthony Senyard present a detailed insight into defect reports and fixes of the OSS project Debian. Their chapter, 'A Statistical Analysis of Defects in Debian and Strategies for Improving Quality in Free Software Projects', provides striking insights into the publicly available data generated by the bug-tracking systems of OSS projects. In doing so, their work highlights the unique research opportunities that these data offer for studying the development processes of OSS projects. The case of Debian – a Linux distribution – shares in common with many OSS projects that its bug reports, fixes and feature requests are governed by tracking systems. One by-product of these tools is that the whole development history is extremely well documented. Based on over 7000 defect reports concerning a number of core packages of the Debian project recorded between 2001 and 2003, Michlmayr and Senyard are able to both describe and evaluate the OSS development process of the project in detail. Their analysis establishes that while the defect submission rate stays constant over the time period in question, the defect removal rate has decreased, a finding hinting at the possibility of an overstretch – or a capacity constraint – in the programmer base. Furthermore, the chapter highlights the increasing difficulties in prioritizing defect reports as projects increase in size, and paradoxically, as they become more successful. Finally the chapter points to opportunities for improving the development process through refined tracking systems and prioritization schemes.

In 'Coworker Governance in Open-Source Projects', Chapter 7, Christoph Lattemann and Stefan Stieglitz deal with aspects of intra-OSS project management. Their work – like that of Michlmayr and Senyard in Chapter 6 – starts from the observation that with the rapidly continuing increase in ongoing OSS projects, the supply of programming power, or developer attention, may become a scarce resource, ultimately leading to competition for programmer capacity between OSS projects, i.e. a programmer shortage. In response, OSS projects pay increasing attention to traditional managerial issues, e.g., to attracting and keeping programming talent, i.e. by engaging in active governance of coworkers. Yet, established concepts of business management may clash with the characteristics of established OSS communities. Lattemann and Stieglitz base their analysis on incentive mechanisms and lifecycle stages of several major OSS projects. They discuss several possible solutions to the coworker shortage issue, ranging from traditional extrinsic motivation measures to the implementation of self-regulation mechanisms based on existing dynamics in OSS communities.

Sandeep Krishnamurthy and Arvind K. Tripathi address in Chapter 8, 'Bounty Programs in Free/Libre/Open Source Software (FLOSS)', a related new phenomenon stemming from the programmer shortage issue. In particular, they document and discuss bounty programs in the management of OSS development. Bounties are schemes where OSS developers are offered financial reward programs for contributions such as creating a specific solution or solving a specific bug, which are becoming more prevalent in OSS development. Bounty programs, by being run as contests that may have deadlines and entry requirements, are one new way to tap into the pool of programmers, attracting developer capacity to a project and shaping the direction of the development process. The contribution by Krishnamurthy and Tripathi is the first to systematically describe this phenomenon. They highlight the benefits and risks of such schemes for the future of OSS and discuss the interaction between these extrinsic motives and the intrinsic motives of OSS developers. Channelling money into a volunteer-programmer environment could potentially speed up development and lead to better-quality work by attracting developers. On the other hand, bounties could end up alienating current volunteer developers, jeopardizing the development process by redefining project priorities and fracturing community structures. Furthermore, this chapter identifies several promising directions for future research on this issue.

Sladjana Vujovic and John P. Ulhøi examine in 'An Organizational Perspective on Free and Open Source Software Development' (Chapter 9), the challenge that OSS-based innovation poses for traditional – profit-driven – innovation models. They establish a set of hybrid constructions between the textbook proprietary knowledge model of innovation and the open source model of innovation, based on non-proprietary knowledge and non-economic motives. These hybrids are related to existing views of collective invention. Vujovic and Ulhøi identify in a review of several major OSS projects the properties and processes underlying the OSS-based innovation activity. They find that OSS-based innovation is neither unmanaged nor unorganized. Rather, the chapter establishes that hierarchies, norms, monitoring and other elements that usually characterize traditional firms also exist in OSS projects, albeit for different reasons than in traditional innovation models. In sum, these features – paired with the detailed documentation of production steps contained in the source code – amount to substantial knowledge production and the associated organizational learning. Furthermore the chapter applies these findings to the coordination and governance of OSS projects and derives managerial implications.

In 'Linux vs. Windows: A Comparison of Application and Platform Innovation Incentives for Open Source and Proprietary Software Platforms', Chapter 10, Nicholas Economides and Evangelos Katsamakas address in a formalized model the interaction of platform versus application developers, thus providing an important new view on innovation activity and OSS. In this context, the platform could, e.g., be the operating system, while the applications would be the

software running on that platform. Their work compares the incentives for these two types of development activities in an OSS environment and a proprietary environment respectively. The innovation activity of either the platform-provider or the applications-provider takes account of the strategic interaction between investments into either activity. Their model shows that ranking of investment levels into the platform is ambiguous, but that the level of investment in the application is larger when the platform is OSS-based compared to a proprietary software-based platform. Furthermore, Economides and Katsamakas establish that the level of investment into the platform depends on the strength of the reputation effects for the developers of the OSS platform, the amount of user-programmers in the population of all users and the cost of adopting the OSS platform system compared to the cost of adopting the proprietary platform.

In Chapter 11, 'The Impact of Entry and Competition by Open Source Software on Innovation Activity', Jürgen Bitzer and Philipp J. H. Schröder provide a further perspective on OSS and innovation by examining the impact of competition on the propensity to innovate. The chapter's starting point is the frequently voiced concern that the emergence and persistence of OSS as a competitor could hamper the innovation activity in the software industry. Two possible channels for such effects exist. First, natural monopoly positions may be part of the industry structure – as is usually argued in other knowledge- and research-intensive sectors such as pharmaceuticals – and hence challenging those structures by market entry may be suboptimal. Second, the competition inherent to OSS, in the forking and branching of projects – situations where projects split into two competing projects ultimately aiming at the same consumer groups – may reduce the overall innovation activity. Bitzer and Schröder address this issue in a simple model of software competition, where firms compete through the technology level of their product, instead of price or quantity; and they present some novel empirical evidence on the issue. The chapter finds that the move from monopoly to duopoly, simulating market entry, has an unambiguous positive effect on innovation activity. Thus competition such as that generated by the entry of OSS, or the forking of projects, promotes innovation.

The book closes with the work of Jin Xu, Scott Christley and Gregory Madey, who provide a promising new perspective by applying social network analysis to the issue of OSS in Chapter 12, 'Application of Social Network Analysis to Open Source Software'. Their chapter describes the topological properties of the social networks formed between the programmers and the groups of programmers in OSS communities. Their study is based on data on OSS communities and developers registered at SourceForge.net and implements a careful distinction of participants according to their role, i.e. ranging from core project leaders to active registered users. Their analysis of the network properties leads to conclusions on developer behaviour, project growth and the diffusion of information. Xu, Christley and Madey show that the developer community displays scale-free network properties and constitutes a self-organizing system, where small-world phenomena also play a role. Their

research underlines the important role of co-developers and active users in bringing about the network properties of the system. They also point to several important directions for the further application of network analysis to the OSS phenomenon.

1.4 CONCLUSION

This book captures in a relatively compact space some of the most important research on the economics of OSS. What puzzles researchers in economics, business science and related fields is that several issues of the OSS phenomenon cannot be explained by off-the-shelf theories. The perplexing particularities such as volunteer programmers, free availability and the emergence of OSS-based for-profit firms create the backdrop for examining central research questions on the OSS phenomenon. We believe that this book contributes to our understanding of these questions, generates new insights and identifies new and promising research ideas. This collection of articles promises to be of interest to academic researchers and graduate students in economics, management science and related fields. Furthermore, we are convinced that this book will be valuable to a wider audience of researchers and graduate students outside the narrow field of economics and management science as well as to policy makers, business leaders and managers in software and related industries.

The book presents three important newly evolved issues in the OSS literature. First, several contributions deal with the interaction of for-profit firms with OSS. These chapters address issues such as the motivation of for-profit firms in their participation in OSS, their relation to OSS communities and the nature of sustainable OSS-based business models. A second important topic presented in this book is the issue of programmer shortage, i.e. the problem that a growing number of OSS projects may exhaust the available pool of voluntary programming capacity. This theme is featured in several chapters that, e.g., present detailed statistical analysis of the Debian project, and observed responses of OSS communities such as managerial activities and tools aimed at attracting programmer capacity to a project. Thirdly, the book presents research that deals with the innovation performance of OSS, addressing issues such as the innovation incentives for further developing an operating system or software applications for an operating system, and the impact of OSS competition on the willingness to innovate in software industries featuring highly concentrated market structures.

REFERENCES

Barcomb, Ann (2004). Open Source and NASA's Mars Rover, in: ON-Lamp.com, http://www.onlamp.com/lpt/a/5053, 2 February.

Bitzer, Jürgen (2004). Commercial versus open source software: The role of product heterogeneity in competition, *Economic Systems*, Vol. 28(4), pp. 369–381.

Bitzer, Jürgen and Philipp J. H. Schröder (2005). Bug-fixing and code-writing: The private provision of open source software, *Information Economics and Policy*, Vol. 17(3), pp. 389–406.

Bitzer, Jürgen, Wolfram Schrettl and Philipp J. H. Schröder (2004). Intrinsic Motivation in Open Source Software Development, Discussion Paper, No. 2004/19, Department of Economics, Free University Berlin.

Feller, J. and B. Fitzgerald (2002). *Understanding Open Source Software Development*, Amsterdam: Addison Wesley Longman.

Festa, Paul (2004). Firefox cutting into IE's lead, in: CNET News.com, available at http://news.com.com/Firefox+cutting+into+IEs+lead/2100-1025_3-5463513.html, 22 November.

Franke, N. and S. Shah (2003). How communities support innovative activities: An exploration of assistance and sharing among end-users, *Research Policy*, Vol. 32(1), pp. 157–178.

Hars, A. and S. Ou (2002). Working for free? Motivations for participating in open-source projects, *International Journal of Electronic Commerce*, Vol. 6(3), pp. 25–39.

Hertel, Guido, Sven Nieder and Stefanie Herrmann (2003). Motivation of software developers in open source projects: An Internet-based survey of contributors to the Linux Kernel, *Research Policy*, Vol. 32(7), Special Issue: Open Source Software Development, pp. 1159–1177.

Johnson, J. P. (2002). Open source software: Private provision of a public good, *Journal of Economics and Management Strategy*, Vol. 11(4), pp. 637–662.

Krishnamurthy, Sandeep (2002). Cave or community? An empirical examination of 100 mature open source projects, *First Monday*, Vol. 7(6), www.firstmonday.org.

Kuan, J. (2001). Open Source Software as Consumer Integration into Production, Working Paper, available at http://ssrn.com/abstract=259648, July 29, 2004.

Lakhani, Karim R. and Robert G. Wolf (2005). Why Hackers Do What They Do: Understanding Motivation Effort in Free/Open Source Software Projects, in J. Feller, B. Fitzgerald, S. Hissam and K. Lakhani (eds), *Perspectives on Free and Open Source Software*, MIT Press, Cambridge, pp. 3–23.

Lerner, J. and J. Tirole (2002). Some simple economics of open source, *Journal of Industrial Economics*, Vol. 50(2), pp. 197–234.

Raymond, E. S. (2000a). Homesteading the Noosphere, Revision 1.22, 2000/08/24, first version 1998.

Raymond, E. S. (2000b). The Cathedral and the Bazaar, Revision 1.51, 2000/08/24, first version 1997.

Torvalds, L. and D. Diamond (2001). *Just for Fun: The Story of an Accidental Revolutionary*, HarperBusiness.

von Krogh, G., S. Spaeth and K. R. Lakhani (2003). Community, joining, and specialization in open source software and innovation: A case study, *Research Policy*, Vol. 32(7), pp. 1217–1241.