Coexisting Intellectual Property Right Regimes: the Case of Open and Closed Source Software

Dissertation

zur Erlangung des akademischen Grades
doctor rerum politicarum (Dr. rer. pol.)

vorgelegt dem
Rat der Wirtschaftswissenschaftlichen Fakultät der
Friedrich-Schiller-Universität Jena
am 29. Juni 2010

von Diplom Volkswirt Sebastian von Engelhardt
geboren am 27. Oktober 1976 in Göttingen

Gutachter

1. Prof. Dr. Andreas Freytag, Lehrstuhl für Wirtschaftspolitik, Friedrich-Schiller-Universität Jena
2. Prof. Dr. Michael Fritsch, Lehrstuhl für Unternehmensentwicklung, Innovation und wirtschaftlichen Wandel, Friedrich-Schiller-Universität Jena
3. Prof. J.D. Stephen M. Maurer, Goldman School of Public Policy and Berkeley Law School, University of California, Berkeley

Datum der Verteidigung: 09.12.2010

To Hanna

Acknowledgments

I would like to express my sincere thanks to those who supported and motivated me during the dissertation.

First and foremost, I would like to thank my doctoral adviser Prof. Andreas Freytag for inspiring support, encouraging supervision and fruitful cooperation as co-author. I am also grateful to Michael Fritsch and Steve Maurer for their work as co-authors and valuable discussions and suggestions. I am thankful for valuable remarks and suggestions by Georg v. Graevenitz, Francesco Rullani and Suzanne Scotchmer.

This dissertation would not have been complete without the support of my colleagues from the Graduate College "The Economics of Innovative Change" and from the Chair for Economic Policy. Especially I would like to thank Angela Münch and Florian Noseleit for helping me with empirical issues. The same holds for Johannes v. Engelhardt. Furthermore, this dissertation would probably not be readable without the help from two native speakers: Julia Marshall and Steve Maurer.

I would like to thank the Klaus Tschira Foundation for financial support by funding the research project "Eine institutionenökonomische Analyse von Open Source und Closed Source Software".

Last, but not least I would like to express my thanks to my parents and friends, who patiently supported me from the very first moment of this PhD project. Here I am especially indebted to Kerrin Klinger.

Contents

List of Figures

List of Tables

List of Abbreviations

BSD	Berkeley Software Distribution
CS	closed source
CSS	closed source software
GNU	GNU's Not UNIX
GPL	General Public License
IPR	intellectual property right
IPRs	intellectual property rights
OS	open source
OSS	open source software

Deutsche Zusammenfassung

Die vorliegende Arbeit beschäftigt sich theoretisch und empirisch mit der Koexistenz von Open Source (OS) und Closed Source (CS) Prinzipien am Beispiel Software. OS und CS sind zwei verschiedene Grundprinzipien in der Definition bzw. dem Schutz geistigen Eigentums – zwei unterschiedliche Regime geistigen Eigentums.

Im Fall von Open Source Software (OSS) ist der Quellcode frei und allgemein zugänglich. Der somit offen gelegte Quellcode ermöglicht es, den Aufbau der Software zu verstehen und diese zu verändern. OSS ist ein öffentliches Gut, das von vielen unabhängigen Entwicklern und Firmen gemeinschaftlich produziert wird. Closed Source Software (CSS) beruht dagegen auf dem Prinzip von exklusivem geistigem Eigentum und wird von einzelnen Firmen produziert, die den Quellcode nicht allgemein zugänglich machen.

OSS hat sich in den letzten Jahren stark verbreitet und wird inzwischen von zahlreichen Firmen der IT-Branche genutzt. Gleichwohl haben die OS-Prinzipien die traditionellen CS-Prinzipien nur zum Teil ersetzt, so dass OSS und CSS koexistieren. Diese Koexistenz ist darauf zurückzuführen, dass es sich bei OSS und CSS verfügungsrechtstheoretisch um zwei second-best Lösungen handelt.

In der vorliegenden Arbeit werden die beiden koexistierenden Regime geistigen Eigentums – OS und CS – unter folgenden Fragestellungen analysiert:

- Wie sind OSS Aktivitäten weltweit verteilt?
- Welchen Einfluss haben länderspezifische institutionelle und kulturelle Faktoren auf die OSS Aktivitäten?
- Inwiefern ist die Koexistenz von OS und CS Prinzipien hinsichtlich geistiger Eigentumsrechte verfügungsrechtstheoretisch zu begründen?

- Wie lässt sich die ökonomische Rationalität von OS und CS basierten Geschäftsmodellen unter Berücksichtigung der verschiedenen OS Lizenztypen modelltheoretisch darstellen?
- Wie sind Marktgleichgewichte mit OS und CS Firmen wohlfahrtsökonomisch zu bewerten und welche Auswirkungen haben hier staatliche Eingriffe?
- Wie unterscheiden sich stark OSS basierte zu stark CSS basierten Startup-Firmen bezüglicher ihrer Gründungscharakteristika?

Im Kapitel 2 wird die weltweite Verteilung von OSS Aktivitäten untersucht. Dazu werden Daten von registrierten OSS Entwicklern bei SourceForge ausgewertet, die vom "SourceForge.net Research Data Archive" zur Verfügung gestellt wurden. Mit Hilfe der angegebenen Email-Adressen, der jeweiligen Zeitzone und der teilweise gespeicherten IP-Adressen konnten dabei 94% der 2006 registrierten Entwickler ihren jeweiligen Ländern zugeordnet werden. Zusätzlich wurden Informationen über die Anzahl der geposteten Forum-Messages ausgewertet, um die Aktivität der Nutzer einschätzen zu können. Das Ergebnis zeigt, dass die relativen OSS Aktivitäten (Messages pro Einwohner) global ungleich verteilt sind. Dies ist auch dann noch der Fall, wenn das jeweilige BIP bzw. die Anzahl der Internet-Nutzer berücksichtigt wird.

Im Kapitel 3 wird daher der Einfluss von länderspezifischen kulturellen und institutionellen Faktoren auf diese globale Ungleichverteilung empirisch analysiert. Dabei wird ersichtlich, dass die OSS Aktivitäten eines Landes stark von kulturellen und institutionellen Faktoren bestimmt werden. Im einzelnen zeigt sich, dass Gesellschaften, die offen gegenüber wissenschaftlichem Fortschritt, von einer Kultur des Individualismus und der Selbstbestimmung geprägt sind, sowohl mehr (aktive) OSS Entwickler pro Einwohner als auch ein höheres OSS Aktivitätsniveau aufweisen. Des weiteren übt auch der Faktor Sozialkapital – gemessen als "interpersonal trust" – einen positiven Einfluss aus. Darüber hinaus sind eine niedrige Regulierungsdichte und ein guter Schutz von geistigen Eigentumsrechten förderlich für OSS. Letzteres verweist darauf, dass OSS zwar die Verwendung von Urheberrecht in seiner traditionellen Form ablehnt, sich aber dennoch auf das Urheberrecht stützt. So sind z.B. die OSS Lizenzen auf die Durch-

setzbarkeit von geistigen Eigentumsrechten bzw. von Lizenzvereinbarungen angewiesen. Dies trifft insbesondere auf OSS Aktivitäten zu, die im Umfeld von OSS Geschäftsmodellen angesiedelt sind. Die Bedeutung von OSS Geschäftsmodellen spiegelt sich auch im Einfluss der Regulierung von wirtschaftlichem Verhalten wieder.

Das Kapitel 4 betrachtet CSS und OSS als unterschiedliche Regime geistigen Eigentums. Die hier vorgenommene verfügungsrechtstheoretische Analyse erklärt die Koexistenz von OSS und CSS mit ex-post Transaktionskosten. Denn bezüglich des Quellcodes sind einige Verfügungsrechte de facto nicht separierbar, was zur Dichotomie von OSS und CSS führt. Da eine optimale Aufteilung der Rechte nicht möglich ist, müssen sich die Individuen für eine der beiden zweitbesten Lösungen OSS und CSS entscheiden. Letzteres basiert auf dem Prinzip exklusiver Verfügungsrechte und ermöglicht eine direkte Kontrolle und Steuerung der Produktion in Firmen (Hierarchie). Allerdings können dabei nicht alle möglichen Kooperationspartner einbezogen werden. Dagegen ist die große Anzahl der Beteiligten die Stärke von OSS, hier wird konsequenterweise auf exklusive Verfügungsrechte weitgehend verzichtet und die Kontrolle nur durch passive Entscheidungsrechte ausgeübt. Rationale Akteure entscheiden sich für OSS oder CSS je nach individueller Situation hinsichtlich ihres Ressourcenzugangs, der Verwendungsmöglichkeit des Quellcodes und den jeweiligen Marktgegebenheiten. Beide Prinzipien sind verfügungsrechtstheoretisch pragmatische second-best Lösungen, mit spiegelbildlichen Vor- und Nachtteilen. Es ist daher naheliegend zu vermuten, dass es gesamtwirtschaftlich Vorteilhaft ist, wenn sich im Markt eine Koexistenz von OS- und CS-Prinzipien etabliert. Die Frage ist aber, ob sich so eine Koexistenz auch etabliert. Die folgenden Kapitel gehen dieser Frage nach und analysieren OS und CS als kommerzielle Strategien für Firmen.

Das Kapitel 5 analysiert in einem formalen Modell das Zusammenspiel zwischen Institutionen und dem strategischen Verhalten der kommerziellen Akteure. In einem allgemeinen, zweistufigen Oligopolmodell (Cournot) konkurrieren OSS Firmen mit CSS Firmen. Beide Typen von Firmen bündeln Software mit Komplementärgütern (Dienstleistungen, Hardware), wobei die OSS Firmen den Code der Software gemeinschaftlich nutzen. In diesem Modellrahmen kann gezeigt werden, dass einfache “neoklassische” Annahmen ausreichen, um die OSS Entwicklung von Firmen rational zu

erklären. Gewinnmaximierende Unternehmen entwickeln OS Code, selbst dann, wenn kein Code durch eine "nicht-kommerzielle" Community (bestehend aus Hobby-Programmierern etc.) bereitgestellt wird. Allerdings ist in diesem Kontext die verwendete OSS Lizenz von entscheidender Bedeutung: während bei restriktiven Lizenzen Firmen bei allen Parameterkonstellationen OSS entwickeln, führen liberale Lizenzen – die die Vermischung von CS und OS Code erlauben – in vielen Fällen dazu, dass Firmen bereits gegebenen OS Code verwenden aber selbst keinen Beitrag zur OSS leisten. Hier ist es gewinnmaximierend, die 'öffentliche' OSS als kostenlosen Input bei der Produktion 'privater' CSS zu nutzen. So produziert keine einzige Firma OSS, denn alle Firmen veröffentlichen ihren Code nur als CSS.

Das Kapitel 6 untersucht, aufbauend auf dem oben genannten Modellrahmen, die implizierten wohlfahrtsökonomischen Aussagen und diskutiert die Notwendigkeit und Wirkung wirtschaftspolitischer Maßnahmen. Dazu werden sowohl der sich unter den Bedingungen des freien Markteintritts etablierende Mix aus OSS und CSS Firmen, als auch deren jeweiliger Marktanteil berechnet. Diese Marktergebnisse werden mit dem wohlfahrtsoptimalen Mix sowie reinen OSS bzw. CSS Industrien verglichen. Es zeigt sich, dass in den allermeisten Fällen ein Mix aus OSS und CSS Firmen einer Industrie mit nur einem der beiden Prinzipien vorzuziehen ist. Allerdings ist die Anzahl der OSS Firmen im Gleichgewicht bei freiem Marktein- und -austritt im Vergleich zur wohlfahrtsoptimalen Lösung zu groß. Die nun folgende Analyse möglicher staatlicher Eingriffe zeigt, dass eine Bereitstellung von OSS oder die direkte Förderung von OSS Projekten eine positive Wirkung hat. Eine Subventionierung von OSS Firmen sollte jedoch unterbleiben. Die optimale Intervention des Staates im Modell ist sogar entgegengesetzt: OSS Firmen werden besteuert und der Erlös für eine Subventionierung der CSS Firmen verwendet.

Im Kapitel 7 erfolgt eine empirische Untersuchung zur Wahl von OSS bzw. CSS basierenden Geschäftsmodellen von Gründungen in der deutschen IT-Branche. Dazu wurde ein webbasierter Fragebogen entwickelt und 6.000 Firmen kontaktiert, von denen sich über 700 an der Umfrage beteiligten. Der verwertbare Datensatz umfasst die Antworten von 680 Firmen. Das Kapitel fragt nach den signifikanten Unterschieden von stark OSS zu stark CSS basierten Gründungsfirmen. Die empirische Analyse zeigt, dass OSS-intensive Start-Ups kleiner sind (weniger Mitarbeiter, weniger Grün-

dungskapital) und daher auch seltener Probleme mit zu geringem Eigenkapital bei Gründung haben. OSS scheint also Firmen-Gründungen zu erleichtern. Dabei tritt auch kein negativer Selektionseffekt auf, denn die niedrigeren Gründungsbarrieren von OSS führen nicht zu einer (durchschnittlich) schlechteren Qualität der Gründungen.

1 Introduction

1.1 The Topic of this Dissertation

Despite differences in detail, the number of conceptually distinct incentives (e.g. patents, prizes, grants, contract research, etc.) that society uses to promote innovation is remarkably small (see Scotchmer, 2004). Against this background, the emergence of fundamentally new "open source" methods for producing software in the 1990s surprised and delighted observers. Open source software (OSS) is marked by free access to the software and its source code, and is developed in a public, collaborative manner. The success of OSS has challenged the conventional wisdom of the role of intellectual property rights (IPRs). The open source principle thus represents a new type of ownership concept for the digital economy.

However, OSS has not completely replaced its counterpart closed source software (CSS), the latter also called proprietary software. As result, OSS and CSS *coexist*, and compete often within the same market. The difference between OSS and CSS is a difference in institutions.[1] OSS and CSS lead to different kinds of "institutional arrangements" (Davis and North, 1971). These coexisting institutional arrangements are distinguishable by their distinct use of copyright law that is codified in the software licenses. The different types of licenses lead to different allocations of IPRs and different governance structures etc.

The present dissertation lays its focus on the coexistence of the two IPR regimes, OSS and CSS. This coexistence is the result of individual institutional choices, the actions under the particular IPR regime, and the result-

[1] Institutions are "the rules of the game in a society, or, more formally, are the humanly devised constraints that shape human interaction" (North, 1990, p 3). They "are made up of formal constraints (e.g., rules, laws, constitutions), informal constraints (e.g., norms of behavior, conventions, self-imposed codes of conduct), and their enforcement characteristics" (North, 1994, p 360).

ing payoffs. Individuals choose to join the OSS community. Developers choose to publish their code as OSS rather than as CSS. Firms choose to use OSS-based or CSS-based business models, etc. Given these micro-level institutional choices, the economic agents choose their action under the respective IPR regime. For example, OSS and CSS firms individually decide how much code to develop. Strategic interactions play a role here, i.e. an OSS firm reacts based on the decisions of the other OSS and CSS firms, and vice versa. The resulting payoffs are such that no IPR regime outperforms the other, but both principles coexist.

These various aspects are analyzed throughout the present study in different ways. The first part (Chapters 2 and 3) of this dissertation focuses on the determinants of the geographic allocation of OSS activities. The institutional and cultural factors that foster individuals' choices to join the OSS regime and become active members of the OSS community are analyzed. One of the findings is that protection of IPR has a positive impact on OSS activities. This supports the view that OSS is a new IPR regime, a new ownership concept, based on intellectual property law. The next part (Chapter 4) therefore presents the rationale for both IPR regimes making use of the property rights approach. It is argued that OSS and CSS are two pragmatic second-best arrangements, with both having assets and drawbacks. Choosing OSS and CSS is rational for individuals and firms. Regarding welfare, the coexistence of OSS and CSS seems to be favorable. Therefore, the third part analyzes the coexistence, including welfare calculations. Chapters 5 and 6 focus on the strategic nature of OSS- and CSS-based business models, taking into account the role of OSS licenses (restrictive or liberal). The equilibrium ratio of OSS and CSS firms (mixed industry) is analyzed. This is followed by a welfare analysis of mixed and pure industries and of possible government interventions. According to the model, there can be a lock-in with only OSS firms, or only CSS firms. This points to the importance of entry. So finally, the entry of firms with OSS and CSS business models comes into focus. The last section concentrates here on start-ups with OSS-based versus CSS-based business models. Chapter 7 analyzes which characteristics of German ICT start-ups can explain their choice between the two IPR-regimes. The dissertation ends with a summary and outlook in Chapter 8: after the summary of the research results, this chapter discusses possible policy implications and further research aspects.

The remainder of the present chapter first offers a short introduction into OSS versus CSS (Section 1.2) and then gives an overview of the state of economic research (Section 1.3). This literature review ends with a short note on the contribution of this dissertation.

1.2 Open versus Closed Source Software: Two Intellectual Property Right Regimes

1.2.1 A Brief History of Open and Closed Source Software

In the early days, software was not a single product but more or less a tool to run the computers. Hence, revenue was created by selling computers, and the hardware vendors delivered software for free. Although some firms were selling the service 'code writing', there was no market for ready-made software products, so-called 'software packages'.

This picture started to change in the late 1960s, when entrepreneurs realized the opportunity to sell their software to more than one customer, hence to treat it like an ordinary mass-marketable product. This new concept diffused, and finally, in the 1980s, the mass publication of packaged software by independent software vendors was established. Meanwhile the U.S. hardware producers – except IBM[2] – withdrew from software (Steinmueller, 1996, p 31 ff.). This rise of the software industry went with increasing concern about the protection of exclusive intellectual property rights. At least since the amendment of U.S. copyright law in 1980, copyright was used to protect intellectual property rights with respect to computer programs. Based on this legal ground, 'proprietary', i.e. closed source, business models were established. As such, the early independent software vendors invented the CSS-based business models, but were also the driving force for establishing copyright protection for software. Thus, they also induced a change on the level of formal institutions. The industry transition to CSS led to some attempts to preserve the 'free' programming culture based on so-called hacker ethics. The most important attempt was the foundation

[2] A history of IBM's software licensing strategies, which reflect the paradigm shift to CSS and then to a balance between OSS and CSS, can be found in Campbell-Kelly and Garcia-Swartz (2009).

of a project called GNU (GNU's Not UNIX). GNU was founded in 1984 by Richard Stallman, who worked at MIT from 1971 to 1984. Stallman was dissatisfied with the rise of the closed source principle, namely with its consequence for the use of UNIX.[3] Therefore he designed and introduced the GNU General Public License (GPL), nowadays the most popular type of open source license. The basic idea of the GPL was to use "copyright law, but flips it over to serve the opposite of its usual purpose: instead of a means of privatizing software, it becomes a means of keeping software free" (Stallman, 1999, p 59). Thus, the GPL was created in order to preserve a certain programming culture of free software and hacker ethics.

This changed the level of institutionalization by transferring some norms of the hacker ethics—i.e. informal institutions—into a formal institution, namely the GPL. With the GPL Stallman invented a new concept of copyright-based ownership: the so-called 'copyleft-principle'. Although Stallman was not motivated by commercial aspects and the creation on the GPL was an act of ideology, Stallman's transformation is an economic success story. For example, all firms with business models build on Linux are based on a GPL-protected software. After the institutionalization of the OSS principle by Stallman and others, several entrepreneurs created business models based on the OSS principle.

So nowadays, firms and individual programmers have an institutional choice whether they use OSS, CSS or both. The next section describes the institutional differences between OSS and CSS. Section 1.2.3 then provides a brief explanation of OSS-based business models.

1.2.2 Open versus Closed Source Principle

Software is traditionally protected by copyright[4] (Graham and Somaya, 2004), and the copyright-based license agreements define the transfer of the intellectual property rights. The crucial feature which distinguishes OSS from CSS is the scope of rights transferred by the OSS vs. CSS licenses.

[3] At this time, UNIX was the most powerful operating system. In the 80s firms started selling incompatible, closed source versions of UNIX.

[4] The discussion about so-called software patents is beyond the scope of this dissertation. For the topic of software patents see e.g. Blind et al. (2005); Hall and MacGarvie (2006); Lerner and Zhu (2007); Bessen and Hunt (2007, 2004); Kahin (2004); Pilch (2004).

Technically this is mirrored by the question of whether there is general access to the source code or not.

The source code is the human-readable recipe of a software program: it is the program code written in a programming language. To run the program on a computer, the source code has to be compiled, i.e. transformed to a (only machine readable) binary code. CSS vendors like Microsoft typically transfer their software only as binary code. They sell the right to use the software, with the scope of legal usage defined by the respective CSS license. For example, copying is not allowed, and this is enforced by law and backed by technical solutions like copy protection. Furthermore, users do not have the right to change the software, and they are also unable to do so as they have no access to the source code.

In contrast to this, an open source code enables users to copy the program code, to understand how the software works, and to change it. Thus, OSS is based on a principle of openness, which is codified in the copyright-based OSS licenses. These OSS licenses permit users to read, modify, improve, and redistribute the code under certain conditions. These conditions vary within a wide range. Liberal licenses—also called 'public' licenses—allow, for example, the use of the open source code to produce CSS. More restricted licenses confine the scope of usage, mainly to ensure that the open source code stays open source.

OSS and CSS licenses differ in the scope of transferred rights. The two IPR regimes thus yield different allocations of IPRs. This has further implications. The definition of ownership of the source determines the governance structures: The principle of CSS is to hold exclusive rights regarding the source code. Closed source code is thus an exclusive asset, and CSS-products are typically developed within single firms. In this hierarchical structure based on exclusive ownership of assets, coordination is achieved by giving orders. Conversely, the source code of OSS is a shared asset. OSS is developed by a decentralized but nevertheless well-organized community. A complex system of rules has emerged to govern OSS development. There exist some hierarchical elements with respect to decision rights, but no one can give orders to other OSS developers. We will come back to the governance structures of OSS in Section 1.3.2 and 4.3.3.2.

The OSS community consists of thousands of volunteers who develop software, often without direct monetary reward. Additionally, more and

more profit-seeking firms engage in OSS development, thus paying programmers to develop OSS code. Large companies as well as small- and medium-sized enterprises use OSS-based business models.

1.2.3 Open Source Business Models

Almost all of today's high tech products are computerized. While this is most obviously true for application software (e.g. games), the point increasingly extends to hardware like cell phones and DVD players. In these industries, a product's quality—and hence consumer appeal—often depends sensitively on the software it contains. Before the 1990s, companies usually developed this as CSS in-house. Since then, however, companies have increasingly turned to shared open source code instead.

Thus, in many markets software is sold and/or used bundled with other goods and services. Consequently, open source (OS) business models are based on these complementary products, as the OS code itself can not be a profit center (Maurer and Scotchmer, 2006, p 289, 290ff). These complements can be hardware like servers or cell phones, premium versions of the software, or different kinds of service like maintenance etc. The following examples provide some idea of the range of OSS business models:

- Many different products – washing machines, mobile phones, flat-screen televisions etc. – are controlled by *embedded software*. Such embedded software can be OSS. Examples of hardware running embedded Linux are Amazon's Kindle, Cisco's MDS and Nexus data switches, Linksys's WRT54G W-LAN router, different Motorola, Nokia, and Panasonic mobile phones, Philips's LPC3180 microcontroller, the TomTom GPS navigation systems, and various LG, Panasonic, Samsung, and Sony LCD and plasma televisions. The most recent example of embedded OSS is Android. Android is a Linux-based software stack (operating system, middleware and key applications) for mobile devices. Acer, Barnes & Noble, Dell, HTC Corporation/Google, Lenovo, LG, Motorola, Samsung, and Sony Ericsson all manufacture and sell products that come Pre-installed with Adnroid.

- Firms in the software industry typically sell a stack of software and services. So-called system integrators even sell a stack of hardware,

software, and services (Riehle, 2007). For example, IBM is selling several servers with pre-installed Linux like the Red Hat Enterprise Linux or SUSE Linux Enterprise Server operating system. This is often based on a collaboration between IBM and the respective Linux distributor. Red Hat, Novell's SUSE and other Linux-*distributors* make money with ready-to-install 'distributions' and the corresponding services like support and maintenance. Such distributions consist of a large collection of well-matched OSS applications, often bundled with further CSS for 'enterprise class' premium versions.

- Internet-based businesses like webhosting and webservices have a high share of OSS-usage. Most web servers are driven by an OSS "Lamp Stack" software suite that includes a Linux operating system, Apache web server, MySQL database, and PHP/Perl/Python programming languages. Development is supported by corporations like Novell, IBM, Oracle, and Borland who then bundle Lamp with their proprietary hardware and software. Small web developers also use Lamp in their businesses and contribute code back to the project.

Furthermore, firms with OSS-based business models have joined several projects and consortia. An example is the Open Handset Alliance, a business alliance of 65 firms for developing open standards for mobile devices, namely the above-mentioned Android. Another important consortium is the Open Source Development Labs (West and Gallagher, 2006). The Open Source Development Labs consist of a wide range of Linux-related providers of hardware, software and services. Table 1.1 provides an overview of the members and their motivations.

Several authors have set up taxonomies of OSS business models. See for example Ghosh et al. (2002a), Fink (2002, Chapter 11), or Daffara (2007). These typologies are of interest for practitioners like managers or for researcher who do detailed empirical research on firms with different OSS business models. However, the different typologies are not of interest for the purpose of the dissertation at hand. We will therefore not discuss these taxonomies further but keep to the following definition that subsumes all the different OSS business models: *Firms with OSS business models generate revenue by selling products (goods or services) that are complements to the freely-accessible OSS.*

Table 1.1: An Example for Commercial OSS: the Members of the Open Source Development Labs

Category	Companies	Motivation
Computer systems vendor	Dell, Fujitsu, Hitachi, HP, IBM, NEC, Sun	Replacing proprietary Unix in computers with shared Linux
Telecommunications vendor	Alcatel, Cisco, Ericsson, NEC, Nokia, NTT, Toshiba	Replacing proprietary Unix with Linux in telecom equipment
Microprocessor producer	AMD, Intel, Transmeta	Enter Unix market using Linux
Linux distributor (server and desktop)	Miracle Linux, NEC Soft, Novell, Red Hat, SuSE, Turbolinux	Sell Linux distributions and services
Embedded Linux distributor	LynuxWorks, MontaVista, TimeSys, Wind River	Design Linux into custom products for customers
Linux support company	VA Software, Linuxcare, LynuxWorks	Sell Linux services
Software developers	Computer Associates, Trolltech	Adapt proprietary applications to Linux

(Source: West and Gallagher, 2006)

1.3 Economic Research on Open Source

During the first decade of the 21st century, the new intellectual property paradigm of OSS (Maurer and Scotchmer, 2006) was attracting more and more interest by economists. In 2000 and 2001 the working-paper versions of three contributions were published, each of them representing one branch of research on OSS. First, the motives of volunteers participating in OSS (Lerner and Tirole, 2000). Second, the coordination of these contributions, hence the governance structures of OSS (Weber, 2000). And third, the impact of OSS on market outcomes and competition (Mustonen,

2001). These three branches were later supplemented by two more recent research aspects: OSS and firms, and open source beyond software. The following sections provide a short overview of each of these branches.

1.3.1 Intrinsic and Extrinsic Motives of Developers

Probably the most famous research question regarding the economics of OSS was asked by Lerner and Tirole (2000, 2002): "Why should thousands of top-notch programmers contribute freely to the provision of a public good?" Lerner and Tirole emphasize the role of extrinsic motivation, namely the acquisition of a reputation-signal. They separate this aspect into two different incentives: career concern incentives, referring to future job offers or access to venture capital, and the ego gratification incentives, which stems from a desire for peer recognition. This seminal article inspired further research on the motives of OSS contributors, analyzing extrinstic as well as intrinsic motives. An overview of this research can be found in Rossi (2006). Today, the consensus is that a mix of extrinsic and intrinsic motives explain the behavior of unpaid[5] OSS developers.

Most of the research on OSS developer motivation consists of empirical studies. These surveys indeed report extrinsic motives like peer recognition and reputation within the community, self-marketing, and career-related motives like the improvement of programming skills and reputation signals (Lakhani and Wolf, 2005; Hertel et al., 2003; Ghosh et al., 2002b; Hars and Ou, 2002; Lakhani et al., 2002). However, most of these surveys find intrinsic motives ranking higher than extrinsic ones. Lakhani and Wolf (2005) find that enjoyment-related intrinsic motivations in the form of a sense of creativity are more important than extrinsic motivations. According to Lakhani et al. (2002) the two top-ranked motives are 'intellectually stimulating' and 'improves skill'. Ghosh et al. (2002b) find that the most important reasons why developers have joined and stay in the OSS community are that they want to learn and develop new skills, and that they want to share their knowledge and skills with other software developers.

[5] Some OSS developers are paid by firms for developing OSS. In such a case their motivation is trivial. The question why firms pay developers to do OSS refers to the topic of firm engagement in OSS (see below).

It is important to note that most of the empirical studies on OSS motives are surveys. Thus the results reflect what the OSS developers report as being their most important motives but do not take into account the importance of the developers, their effort levels, etc. An exception is the article by Hars and Ou (2002). They connect the reported motives with the individual effort and find that, although intrinsic motivations play a role, external motives have greater weight, see Table 1.2. Hars and Ou (2002) also point out, that different types of OSS programmers exist. For example students and hobby programmers are more internally motivated than professionals.

Table 1.2: Motivations and Effort of OSS Programmers

	All		Students and hobby programmers		Salaried and contract programmers		Programmers paid for open-source development	
Motivation	**Percent**	**Corr. with effort**	**Percent**	**Corr. with effort**	**Percent**	**Corr. with effort**	**Percent**	**Corr. with effort**
1. Internal								
Self-determination	79.7%	0.072	81.8%	-0.015	92.6%	-0.303	61.5%	0.221
Altruism	16.5%	0.192	24.2%	0.356	11.1%	0.061	7.7%	-0.163
Community identification	27.8%	0.116	36.4%	0.361	18.5%	-0.130	30.8%	-0.307
2. External								
2.1 Future rewards								
Selling products	13.9%	0.363	6.1%	0.011	3.7%	0.488	53.8%	0.304
Human capital	88.3%	0.139	96.9%	0.080	88.5%	0.073	84.6%	0.065
Self-marketing	36.7%	0.317	33.3%	0.206	29.6%	0.208	69.2%	0.424
Peer recognition	43.0%	-0.021	42.4%	-0.023	48.1%	-0.145	46.2%	-0.178
2.2 Personal need	38.5%	0.304	36.4%	0.301	38.5%	0.186	38.5%	0.328

(Source: Hars and Ou, 2002)

1.3.2 Governance Structures and Licenses

Aside from the question of motives, Weber (2000) asks how the OSS developers "coordinate their contributions on a single 'focal point'?" (Weber, 2000, p 5). Research on OSS has thus to understand how the implications of the complexity of large OSS projects like Linux are managed. Consequently, in Weber (2004b) he describes collaborative methods in the context of developing OSS. This points to the institutions of OSS, including organizational issues and governance structures, the role of hacker-ethics and the role, choice and rationale of OSS licenses.[6]

What kind of organization OSS projects represent and how they are governed is a question that is widely discussed nowadays Markus (2007) offers a sound survey and a synthesis of this literature. Markus defines OSS governance as "the means of achieving the direction, control, and coordination of wholly or partially autonomous individuals and organizations on behalf of an OSS development project to which they jointly contribute" (Markus, 2007, p 152). Markus (2007) rightly points out that two branches of research on OSS governance can be distinguished. Some scholars analyze OSS as a new, distinct but unitary organizational form which can be differentiated from CSS development (Raymond, 1998), characterized as a private-collective model (von Hippel and von Krogh, 2003; Osterloh and Rota, 2007), and so on. Others emphasize the different types of OSS governance mechanism, or focus on one of these mechanisms specifically. An overview of the governance mechanisms is provided by (de Laat, 2007). He groups the main tools of OSS governance into six categories: modularization, division of roles, delegation of decision-making, training and indoctrination, formalization, and authority versus democracy. What follows contains examples from both types of literature.

Let us start with OSS as a unitary organizational form. Very common in organizational theory is the tripartite division of types of organization into 'market', 'firm' and 'network' (see Table 1.3). In this context, OSS projects are mostly characterized as networks. An exception is the interpretation of Demil and Lecocq (2006). They argue that OSS projects differ from networks in that they do not require long-term relations, have no mechanism

[6] Gehring (2006) and Lessig (2006, 1999) interpret the code itself to be an institution. However we do not follow this view here.

Table 1.3: The three Types of Organization

Type of Organization	Type of Coordination	Coordination Instrument
Market	Competition	Price
Firm	Hierarchy	Order
Network	Cooperation	Consensus

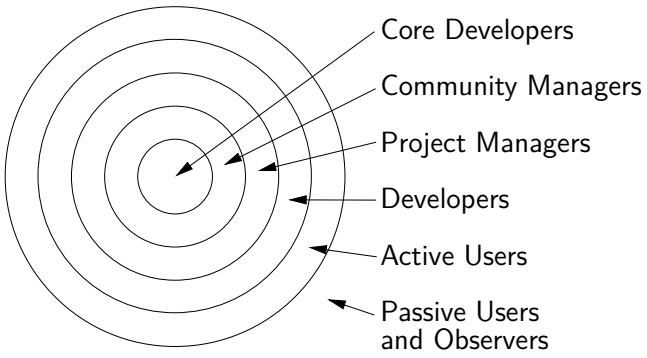
(Source: Brand and Schmid, 2005)

to restrict access, etc. Also Garzarelli (2003) points out the organizational uniqueness of OSS, arguing that its organizational characteristics can be explained by a combination of the organizational theory on clubs with the theory of professions. However, many authors (e.g. Brand and Schmid, 2006, 2005; von Hippel, 2005; Iannacci and Mitleton-Kelly, 2005; Benkler, 2002) interpret OSS projects as (special) networks. Based on a case study on the KDE project[7] Brand and Schmid (2005, 2006) find that OSS combines the coordination mechanism of networks with elements of hierarchy, the latter typically associated with firms.

These hierarchical elements of OSS organizations are mainly based on decision rights and the tasks of certain developers. The basic structure of these hierarchies is often labeled the 'onion layer' model, see Figure 1.1. (See also Section 4.3.3.2 and Jensen and Scacchi, 2007; Crowston et al., 2006; Wendel de Joode et al., 2003, pp 18,19.) The career of developers within projects (a participant becomes project leader or specialist etc.), i.e. their movement into the core of the 'onion' is studied by Giuri et al. (2008). The acceptance of the 'onion'-hierarchies and the authority of e.g. project leaders are based on the meritocratic norms of the OSS community, rooted in hacker-ethics (O'Mahony and Ferraro, 2007). Johnson (2006) and Lee and Cole (2003) emphasize the importance of peer review processes in the control structures of OSS. In this context von Krogh et al. (2003) point to the role of extrinsic motives and incentives like reputation and signaling.

[7] KDE (K Desktop Environment) is an open source graphical user interface (GUI). Together with the GNOME desktop it is likely the most-known desktop environment (and development platform) for Linux and Unix workstations (Webpage: www.kde.org).

Figure 1.1: The 'Onion Layer' of OSS Projects



(Source: Jensen and Scacchi, 2007)

Several authors underline that the evolution of OSS projects over time has implications for their organizational forms (Sadowski et al., 2008; Lattemann and Stieglitz, 2005; Schweik and Semenov, 2003; Wynn, 2003). Typically, OSS projects start with one or only a few developers who coordinate via direct communication based on trust. With the growth of the project size, more and more official coordination structures are implemented. Finally the projects characterized by well-defined roles (code-tester, release-manager, core-developer etc.) combined with a decentralized and modularized organization structure. For example, Crowston and Howison (2005) have analyzed 120 project teams from SourceForge and find that growing projects become more modular, with different people responsible for different modules. Langlois and Garzarelli (2008) explicitly focus on modularity in open source collaborations, including OSS. They argue that in such collaborations the division of labor is coordinated through voluntary exchanges of effort rather than of products.

In his synthesis of research on OSS governance, Markus (2007) concludes that three OSS governance purposes are linked to six OSS governance categories, see Table 1.4. According to Markus (2007), OSS governance has to solve collective action dilemmas and coordination problems, and has to create a climate such that developers contribute to the particular

Table 1.4: Relationship between OSS Governance Purposes and OSS Governance Categories

OSS governance purposes vs. OSS governance category	Solve collective action dilemmas	Solve coordination problems	Create a climate for contributors
Ownership rules	x		x
Chartering rules	x		x
Community rules	x	x	x
Software development process rules		x	
Conflict rules and rules about rules		x	x
Information and tools rules		x	

(Source: Markus, 2007)

project (rather than to others). To achieve these goals tools from the six governance categories are used. For example, rules about the software development process and rules about how information will be communicated and managed using certain tools (repositories) both support coordination. Conflict rules as well as meta-rules solve coordination problems but also create a good climate for contributors. The community rules – which determine who can become a member, what roles members can play etc. – clearly support all three purposes. Finally, tools that solve collective action dilemmas and create a contributor-friendly climate belong to the categories of chartering rules and ownership rules. The first refers to statements about the goals of the project, what the software should look like, and so on. The second refers to the use of (intellectual) property law: the formal legal organizational structure (e.g. a foundation) and the type of license etc.

O'Mahony (2003) shows in detail how OSS projects use intellectual property law to protect their work. OSS projects make use of restrictive license terms or trademark registration etc. Additionally, often the copyright, trademark etc. is transferred to a foundation. Such foundations are better suited to enforcing e.g. the license restrictions or protecting the brand of the project.

The type of OSS license is an important institution, as it defines how the code can be used. Some authors analyze the importance of the respective license for the governance of an OSS project. For example, Franck and Jungwirth (2003) argue that the GPL is constructed such that egoistic motives ('rent seeking') do not crowd out altruistic motives ('donation'). Moreover, according to Franck and Jungwirth (2003), the GPL creates incentives for participation for both rent seekers and donators.

Sen et al. (2008) examine how the OSS license type (ranging from very restrictive to very liberal) can be explained by the motivations and attitudes of the OSS developers. They find that intrinsic motivation of challenge (problem solving) is connected with a preference for moderate restrictions, while extrinsic motivation of status (peer recognition) is linked to licenses with least restrictions. Another study on the determinants of OSS license choice is offered by Lerner and Tirole (2005). They first develop a theoretical model and then test the model predictions empirically. According to Lerner and Tirole, OSS projects are more likely to have restricted licenses if they are consumer-oriented (e.g. desktop tools or games) and if they

are developed in a corporate setting. Projects oriented toward developers and/or designed to run on commercial operating systems have less restrictive licenses. Finally less restricted projects tend to attract more developers. A discussion about the relationship between OSS business models and the type of OSS licenses is provided by de Laat (2005). This includes a description of Netscape's experience with different licenses, when the company turned its browser Netscape Navigator into an OSS project – today this OSS project is named Mozilla with its products Firefox and Thunderbird.

The literature on licenses mentioned so far concentrates on OSS licenses only. But some scholars focus on OSS versus CSS licenses. Using a centipede-type game, Polanski (2007) analyzes CSS versus OSS licensing as a mechanism design issue. He models cumulative production, thus sequential production where the outputs of the stages $1 \ldots k-1$ are inputs for stage k. In such an environment, according to Polanski, a public (open source) license is better suited if the project is highly modular and there are significant returns to scale. Bessen (2006) uses a model based on incomplete contracting and the hold-up problem to shed light on the rationale of OSS licensing. The result of his model is that OSS licensing can be more efficient than CSS licensing in the case of complex products like software. According to Bessen, OSS will be mainly used by firms that have complex specialized needs and their own development capabilities. Another paper based on incomplete contracting is D'Antoni and Rossi (2007). They analyze the rationale for liberal versus restricted licenses, hence the BSD license[8] versus the GPL. D'Antoni and Rossi (2007) find that the GPL is superior to coordinate and encourage joint effort by many (possibly small) developers; while the BSD is better suited to generate positive spillovers to other developers when no feedback is required.

1.3.3 Market Outcome and Competition

Mustonen (2001, 2003) was the first to analyze the impact of OSS on competition and market outcome. This was followed by several contributions on this topic. In this context it is useful to distinguish models with non-commercial OSS from those with commercial OSS. In case of the first, all

[8] The license of the Berkeley Software Distribution.

the OSS is provided by the non-commercial community. There can be firms who use this OSS as input, but there is no OSS developed by firms. The second branch focuses on markets where firms contribute to OSS.[9]

Mustonen (2001, 2003) belongs to the branch focusing on non-commercial OSS. He models the interaction between the OSS community and a CSS monopolist. The monopolist is affected by OSS in two markets: the product market and the labor market. Consumers can either buy CSS or use OSS for free. However, both types of software cause implementation costs. Programmers choose to work for the monopolist at a wage that the monopolist sets, or develop OSS and thus build reputations that results future income. In Mustonen's setting, highly talented programmers have incentives to join the OSS community. The basic result is that if the software implementation costs are low, OSS and CSS coexist. The presence of OSS lowers the CSS vendor's monopoly power in both markets. The impact of non-commercial OSS on the outcome of software markets is also analyzed by Casadesus-Masanell and Ghemawat (2006). Their model is inspired by the competition between the operating systems Linux vs. Microsoft's Windows and is a dynamic mixed duopoly of CSS and non-commercial OSS. Casadesus-Masanell and Ghemawat (2006) take into account dynamic effects which yield network externalities: the cumulative output of each operating system (installed base) affects their relative position over time. They find that Windows can survive in the market, hence coexist with Linux, if the installed base effect is strong enough.[10] Additionally, they show that welfare in the mixed case can be smaller than under a Windows monopoly. Economides and Katsamakas (2006) focus on the fact that operating systems like Linux and Windows are platforms. Making use of the theory of two-sided markets, Economides and Katsamakas (2006) compare industry structures based on an OSS platform with those based on a CSS platform. They compare a vertically integrated CSS-industry, a vertically dis-

[9] Clearly, models with commercial OSS also contribute to the topic 'OSS business models'. The distinction is made based on the focus of the respective article. If the main purpose is to analyze the market outcome, then we consider it in this section. If the main purpose is to explain the rationales for OSS business models, then we mention it in the next section.

[10] They assume that in $t = 0$ Windows is perceived more valuable than Linux. Hence Windows has an advantage in the beginning.

integrated CSS-industry, and an industry with an OSS platform and CSS applications. Economides and Katsamakas (2006) provide conditions for each of these industries to have the highest industry profits. They also find that welfare is maximized if the industry is characterized by an OSS platform with different CSS applications. Also in Bitzer (2004) OSS is developed solely by the community. He analyzes a case where a CSS firm faces the emergence of OSS. However, Bitzer (2004) takes into account firms with OSS-based business models. Firms with OSS business models can use the OSS code for free and bear only the costs for producing the complementary products. The CSS firms on the other hand have to bear the costs for both software development and production of complements. Bitzer (2004) uses a Launhardt-Hotelling model set-up and derives the result that product heterogeneity is the crucial factor in this setting. If the heterogeneity between the OSS and CSS based products is sufficiently high, the CSS firm will stay in the market. A model with competition in technological levels rather than in prices or quantities is proposed by Bitzer and Schröder (2007). They find that the chosen technological level is higher in markets with OSS and CSS than in pure CSS markets. The highest technological level is achieved in pure OSS markets.

Sen (2007) models competition in software markets where CSS vendors compete against firms who sell *improved* versions of OSS. The latter represents the business model of OSS 'distributors' as explained in Section 1.2.3, p 7. Firms take a given OSS and improve its usability with support and service (SS). Therefore Sen calls this type of software OSS-SS. Consumers can thus choose to use either OSS for free, or purchase either OSS-SS or CSS. The CSS and OSS-SS firms decide on the usability of their software, while OSS has a fixed, low usability. Furthermore, users differ in their valuation of software usability (a Hotelling's model approach). Sen (2007) takes into account network effects in terms of installed base. Here OSS and OSS-SS users belong to the same installed base, as both use the same software in technical terms. Sen (2007) finds the following results, from which he then also draws management implications. With weak network effects, CSS always have a market share of more than 50%, with its usability and prices being higher than those of OSS-SS. If network effects are high, profits for the OSS-SS vendors are maximized if they offer the same usability as their CSS rivals. In such a case, CSS is driven out of the market.

Firms that do not only use but also develop OSS (commercial OSS) are the topic of Verani (2006) and Schmidtke (2006). In both cases OSS firms develop code and produce complementary products. While Schmidtke (2006) analyzes the impacts of OSS business models (e.g. welfare issues) in a non-differentiated Cournot oligopoly, Verani (2006) uses a duopoly model to analyze under which conditions firms produce more code, under an OSS rather than under a CSS regime. To the best of the author's knowledge, the only contribution that models commercial OSS and analyzes a mixed industry with OSS *and* CSS firms is Llanes and de Elejalde (2009). We will come back to this literature in Chapter 5.

1.3.4 Incentives and Role of Firms

Beside the motives of individual contributors, the engagement of firms is of interest. Here research analyzes the incentives for firms to contribute and the roles firms play within the OSS community.

Dahlander (2007) analyzes the role firms play in OSS projects, distinguishing between projects initiated by firms versus community-initiated ones and high versus low degree of firm participation. He focuses on de novo entrants (new organizational entities are formed) and draws conclusions for the management of OSS-based business models. The fact that an OSS project was founded by a firm rather than by the community has influence on its governance structure (West and O'Mahony, 2008). Governance of community projects is largely pluralistic, while in firm-initiated projects the ultimate decisions are controlled by the company.[11] Furthermore, firm-initiated projects tend to have less restrictive licenses rather than the GPL. Here firms are an origin of more flexibly licensed OSS (Koski, 2005), including the strategy of dual-licensing (Välimäki, 2003) like e.g. an open source basic version and a closed source premium version.

Dahlander and Magnusson (2005) examines how the relationships that firms have to the OSS communities are connected with their way of doing business. He distinguishes three types of strategies. With the "symbiotic approach" the firm and the community gain, as the firm strongly contributes back. If the firm uses a "commensalistic approach" (firm uses input from

[11] This can also be a group of firms founding an alliance.

the community), the firm gains while the community is indifferent. Finally in a "parasitic" firm-community relationship the firm exploits, i.e. uses input without obeying norms, values and rules of the community. Not surprisingly, Dahlander and Magnusson (2005) report that firms who use a more symbiotic approach have more possibilities to influence the community. But such firms have to manage their dual roles of being a profit-seeking firm and part of the community. The competitive advantage of an OSS firm can hence be influenced by the relations it may have with OSS communities. Dahlander and Magnusson (2006) emphasize in this context that in order to successfully cooperate with and gain from the community, firms have to have capabilities and in-house expertise.

Henkel (2009) focuses on the individual developers who establish the link between OSS firms and the OSS community. Here a principal-agent problem might exist, caused by the developer's double allegiance to firm and community. Thus some firms fear the risk of losing intellectual property, etc. Henkel (2009) uses data derived from interviews and a large-scale survey. He finds no evidence of commercially harmful behavior induced by OSS ideology ("Software has to be free" etc.). Also Dahlander and Wallin (2006) emphasize the role individuals play in the attempts of firms to unlock communities as complementary assets. Based on network analysis they show that firms sponsor individuals to act strategically within the OSS community.

Based on data from SourceForge (the leading online depository for OSS projects), Lerner et al. (2006) analyze the kind of projects to which firms contribute. They find that firms tend to contribute more to larger projects that grow faster (in terms of code lines). In their dataset, Lerner et al. (2006) can not find any consistent relationship between the type of OSS license and corporate contributions.

Henkel (2006) is focusing on the incentive for firms to contribute code back to the community even if they are not obliged to do so. The explanation he can draw form his empirical study is that that firms can expect a kind of reciprocal behavior: they receive informal development support from the community which even includes other firms. At the same time firms protect their intellectual property by reveling only parts of their code, only several modules respectively. Henkel (2004) provides a duopoly model of why firms use and contribute to embedded Linux. The two firms

require two technologies (or software modules) but value these technologies differently. As a result, each firm concentrates on producing the software it values most, publishes this as OSS and receives the OSS developed by the other firm. The economic logic of Henkel's model is basically a dual version of the 'exploitation of the great by the small' analyzed by Olson and Zeckhauser (1966); Olson (1971). In their duopoly model, Baake and Wichmann (2004) analyze the rationale of firms to publish parts of their software as OSS. Because of spillovers, publishing code as OSS reduces the firms' coding costs. But OSS encourages entry and thus increases the expenditures required to deter entry.

1.3.5 Open Source Beyond Software

Some authors discuss the possibility to implement the open source (OS) paradigm in areas other than software. This must not be confused with research on online communities beyond OSS, like research on Wikipedia (e.g. Gaio et al., 2009; Ciffolilli, 2003). Here OS beyond software means applying similar OS mechanisms to other industries based on digital goods, i.e. "payoff-relevant bitstring[s]" (Quah, 2003). For example, an important project of the OS movement in genomics-based research is the International Human Genome Project. Laboratories from all over the world jointly collaborate to map and sequence the human genome, with the resulting data deposited into the public domain.

Maurer (2008) discuss if and how OS principles and incentives are suitable for the several stages of the drug discovery pipeline. Allarakhia et al. (2010) examine the mechanisms of cooperative knowledge production and dissemination in OS biopharmaceutical innovation. They analyze about 50 OS initiatives that focus on genomic, proteomic, and systems-based research. Based on this, Allarakhia et al. (2010) develop a two-player game model in order to further analyze the incentives to participate in OS biopharmaceutical initiatives. In her book "Biobazaar", Janet Hope discuss the challenges and implications of applying OS principles to biotechnology. She argues that OS biotechnology would foster competition in the industry that today tends to be dominated by a few powerful players Hope (2008). Henkel and Maurer (2007) discuss the economics of OS synthetic biology, including the consequence of different access and usage rules regarding

the community's Registry of Standard Biological Parts. Finally, Roosendaal (2007) discusses the (legal) problems that occur when commercial companies are invited to join OS projects in biomedics. Such problems are a result of the tension between traditional proprietary regimes and OS approaches in this field.

1.3.6 The Contribution of this Dissertation

This dissertation focuses on the coexistence of OSS and CSS. As OSS and CSS differ in institutions this dissertation is based on the viewpoint of the new institutional economics. The above literature overview shows that most of the research concentrates on the phenomenon OSS. And even in models analyzing the effects of OSS on market outcome and the competition between OSS and CSS firms, the coexistence is exogenously given.[12] It is assumed that e.g. several OSS firms compete with one CSS firm or that an CSS incumbent faces an OSS community. So there is still a lack of research that is able to explain the coexistence endogenously. The present dissertation contributes to this. The fact that OSS and CSS coexist is explained by different individual institutional choices, the actions under the certain IPR regime (including the strategic interactions) and the resulting payoffs. In a nutshell: The choices for OSS and CSS on the micro level and the resulting payoffs are of interest. Neither the one nor the other IPR regime rules out its counterpart because the OS and the closed source (CS) principle have relative advantages and drawbacks. Whenever possible, the analysis uses a general framework such that the results also contribute to OS beyond software.

The following gives a brief overview on the contributions of the different chapters. The data about the world-wide diffrerences in OSS activties of Chaper 2 provide a more accurate geography of the supply side of OSS than any other study. Based on this dataset, Chapter 3 provides the first study on the influence of country specific institutional and cultural factors on the decison to be an active member of the OSS community. Chapter 4 explains the coexistence of the two IPR regimes using a property rights approach. Chapter 5 presents the first general Counrot model analyzing the strategic

[12] An exception is the contribution by Llanes and de Elejalde (2009). For more details on this paper see Section 5.2.

nature of OS vs. CS business models and the role of OS license-type. Based on this, Chapter 6 deepens the analysis by discussing welfare aspects and the consequence of possible government interventions. Chapter 7 presents the first study comparing the characteristics of ICT start-ups with OSS- vs. CSS-based business models.

2 The Geographic Distribution of Open Source Software Activities

2.1 Introduction*

The provision of OSS appears to be a case of a "private provision of a public good" (Johnson, 2002). As the community is often described as global, OSS seems to be a digital public good with a truly globalized private provision. However, apart from anecdotal evidence for the internationality of certain OSS project teams, the question remains how global the OSS community actually is and how the supply side of OSS differs among countries. This has motivated researchers to study the geographical allocation of OSS developers. It turns out that the most OSS developers come from North America and Europe. This result is quite consistent regardless of the method used. The methods to gather information about the geographic origin of OSS developers can be broadly distinguished into two approaches. Some studies are based on survey-data, while other work is based on specific data drawn from code of certain OSS projects such as credit files, mailing lists or data from platforms like SourceForge.

Robles et al. (2001) provide a combination of both types of data collection. In Ghosh (2006); David et al. (2003) and Ghosh et al. (2002b) one can find survey-based information about the origin of OSS developers. Lancashire (2001) provides information about the world-wide distribution of Linux and Gnome developers, based on data collected from the Linux Credit file and in case of Gnome developer-contact information from the project's web-site. The most recent research dealing with the geographic origin of OSS developers is Gonzalez-Barahona et al. (2008). These authors provide a worldwide picture of OSS developers, weighted by population, in-

*This Chapter is based on von Engelhardt et al. (2010)

ternet users and GDP. Gonzalez-Barahona et al. (2008) build on Robles and Gonzalez-Barahona (2006). Robles and Gonzalez-Barahona (2006) use information about the email addresses of registered users and the indicated time-zone to assign developers at SourceForge in 2005 to their countries. However, they were unable to assign 25% to countries, because of the combination of a generic (non-country specific) Top Level Domain like .com with the country unspecific timezone GMT. Robles and Gonzalez-Barahona (2006) develop methods to estimate the geographic allocation of this 25%.

Our work is inspired by Gonzalez-Barahona et al. (2008) and Robles and Gonzalez-Barahona (2006), but proceeds along two lines: First, we do not have to estimate any geographic origins, since we can directly assign 94% of all developers registered at SourceForge in 2006. We make use of relevant information obtained from email, time zone and the Internet Protocol addresses. Combining these, we are able to assign 1.3 million developers to their countries without the need to estimate geographic origin. We cross-checked the results which delivers an indicator for the validity of each of our methods. Second, we provide information about how active each developer is. With individual data about the number of posted messages we have a good proxy for activity. We can thus distinguish active from non-active (but nevertheless registered) developers, and we are able to show the worldwide allocation of OSS *activities*. Information about activities are important, since members of the OSS community differ in their effort levels, numbers of contributions etc. (see e.g. David and Rullani, 2008).[1] With the active developers and activity, our study can show a more accurate geography of the supply side of OSS development.

2.2 The Need for Accurate Data about OSS Activities

Software-development is an important part of each country's ICT-Sector. But without information about OSS activities the picture of the software industry's supply side remains incomplete. For example, workforce and

[1] For further literature on the division of labor within open source projects etc. see among others den Besten et al. (2008); Giuri et al. (2008); von Krogh et al. (2003).

human capital data typically count only the paid labor force and thus ignore the—for the most part unpaid—OSS developers. On the other hand, adding the number of OSS developers to the number of paid jobs is also incorrect for two reasons. First, not all registered developers are active: at SourgeForge, only every fifth member was active in 2006. Second, some OSS developers have jobs in the software-sector which would lead to double-counting. Taking the OSS-activity-level into account can, together with the numbers of the paid software development, provide a more accurate picture.

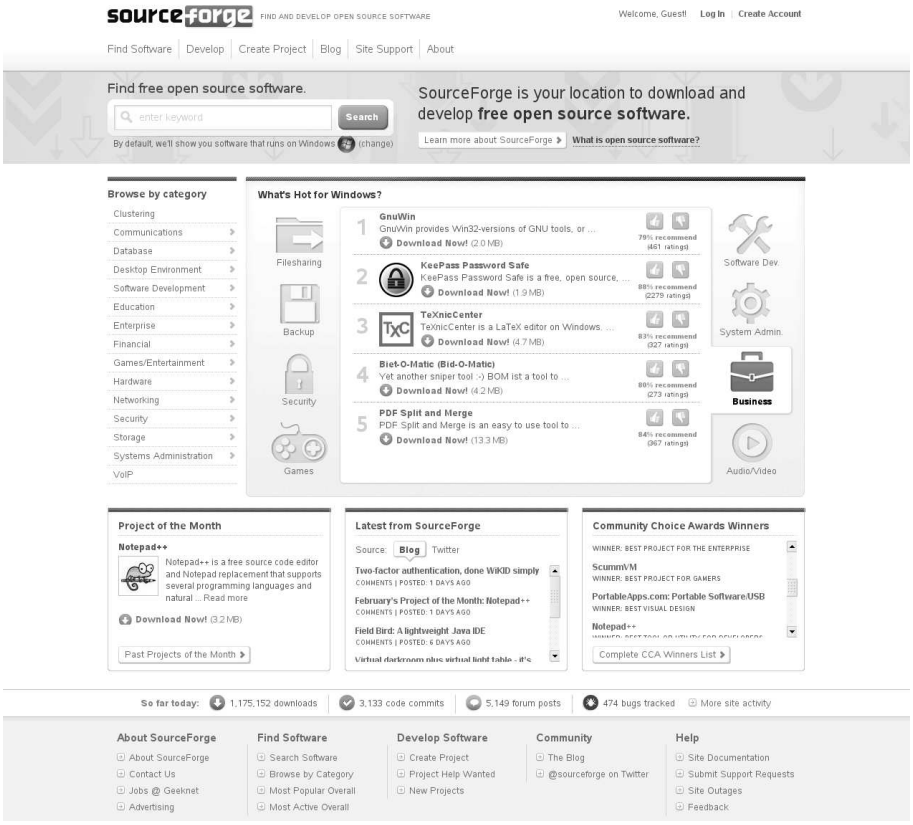
Data including the number of active OSS developers and their activity level by country are important for both, policy makers and businesses. Government competition policy as well as support for OSS development, OSS-based business models and OSS-based start-ups should be based on knowledge of the national human capital and capacity for OSS. This information is also useful for e.g. entrepreneurs contemplating an OSS-based start-up, firms planing to implement an OSS-based business model, etc.

Additionally, a better and more complete picture of software-development accross countries can improve cross-country studies, including aspects of the digital divide. Finally, our dataset can be used for analyzing country-specific impacts on OSS activities. The fact that its magnitude differs among countries points to the institutional and cultural 'embeddedness' of OSS. Therefore, the next chapter uses this dataset to analyze the role of country-specific culture and institutions.

2.3 Data Source and Methodology

SourceForge is an internet platform designed to help developers to control and manage OSS projects. Figure 2.1 shows the start page of SourceForge. SourceForge provides a virtual center where the developers of a certain OSS project can meet, discuss, coordinate tasks, upload new developed codes, etc. SourceForge also records and documents these activities. For example see Figure 2.2 which shows the documentation for committed code deposited in the 'TeXniCenter' project. SourceForge is the largest repository of OSS projects. While finished version of software can be downloaded by

Figure 2.1: Start Page of SourceForge.Net

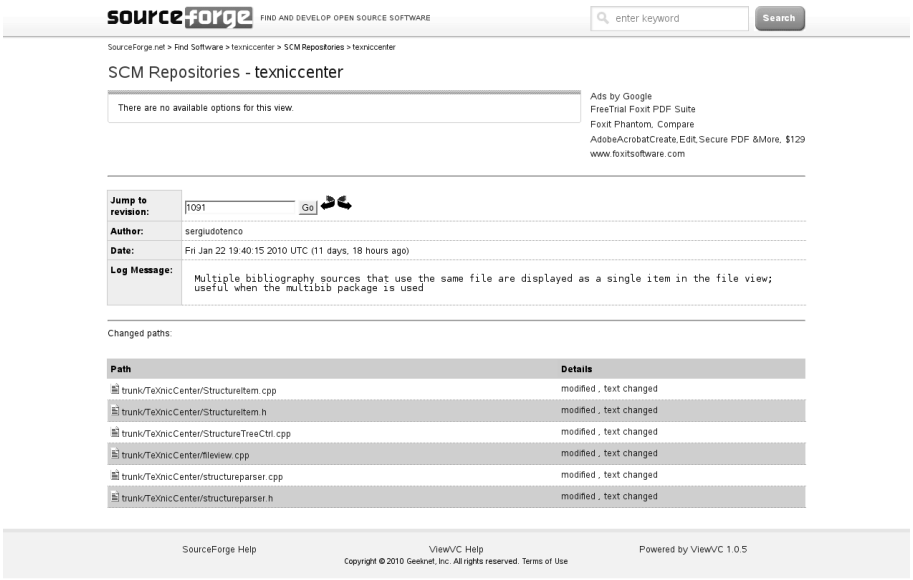


anybody, access to the developer areas requires registration. This includes some personal data including a valid email address.

Research on OSS often uses SourceForge data.[2] Nevertheless, not all OSS projects are hosted on SourceForge. Namely asain regions are particularly under-represented, as they have more local OSS communities (see Gonzalez-Barahona et al., 2008, p 358). However, "from an economic per-

[2]See for example Au et al. (2009); Giuri et al. (2010); David and Rullani (2008); Eilhard (2008); Gonzalez-Barahona et al. (2008); Fershtman and Gandal (2008); Comino et al. (2007); Robles and Gonzalez-Barahona (2006); Lerner et al. (2006); Xu et al. (2006), and Lerner and Tirole (2005)

Figure 2.2: Information about Comitted Code on SourceForge



spective it is useful to examine the distribution of participation in global projects" Gonzalez-Barahona et al. (2008). In this respect, data derived from SourceForge make a good indicator.

We obtained our data about registered SourceForge.net developers from the SourceForge Research Data Archive (SRDA). SRDA is offered by the University of Notre Dame under a special agreement for scientific research (Madey, n. d., see also `www.nd.edu/~oss/Data/data.html`). The database consists of monthly dumps containing parts of the information stored at the SourceForge web-page. The latest dumps containing all of the information needed for our analysis date from the year 2006. Because we are able to identify each user by the user-ID we can compile the indicated email address and time zone, the saved Internet Protocol address and the number of posted messages for each registered developer.

When OSS developers register at SourceForge they have to indicate a valid email address. Additionally, registering developers have the option to change the time zone from the default-value to their specific time zone (e.g. "Europe/Berlin"). Email address and timezone are saved in the 'users'-table of SourceForge. However, the SRDA team stopped including email addresses from user tables in its dumps for privacy reasons starting in October 2006.

The SRDA dumps also contains tables storing the Internet Protocol address of the users who logged into the site. Internet Protocol addresses of registered users can be found in the Tables 'user_ip_dl_auth' and 'audit_trail_users'. The first one consists of information about users who have registered in the respective month. The second table consists of data generated by SourceForge in order to be able to restore the data, i.e. are data used for backups. Here data are saved only when something was changed (data changed/uploaded by a user etc.). Nevertheless, only the SRDA-dumps of July, August, September and October 2006 contain the Tables 'user_ip_dl_auth' and 'audit_trail_users'.

The original 2006 dumps contain approximately 1.4 million datasets. We clean this information by removing duplicates, fake accounts and unreliable data. We then assign to each user his or her geographical origin based on the email address, the time-zone and the IP address:

country coded Top Level Domain (ccTLD) If the Top Level Domain of the respective email address is a country coded Top Level Domain (ccTLD), we can use this information to assign users to countries. For example, emails ending with ".us" will be assigned to the USA, with ".nl" to the Netherlands, or with ".de" to Germany. Thus, our underlying assumption is that each user's ccTLD correctly indicates his or her native country or the country of (long-term) residence respectively.

So-called open ccTLDs present a special problem. This is because *open* ccTLDs are not limited to citizens or firms of the respective countries, but are made available internationally to any interested registrant for a fee (Edelman, 2002). The reason is that global websites find certain Top Level Domains very attractive. For example, ".tv" (for Tuvalu) looks like "television", or ".ws" (Western Samoa) looks like "website". Such open ccTLDs can *not* be used for geograph-

ical identification, they are in fact as TLDs like ".org" or ".com". We therefore exclude all open ccTLDs from the dataset when identifying via country coded Top-Level Domain of the email addresses.

Second Level Domain (SLD) All email accounts with generic TLDs contain information in the form of a so-called second level domain (SLD). For example, "yahoo" the SLD in case of "xyz@yahoo.com". It is possible to identify the location of the SLD domain servers. We therefore manually assign to each of the top 1000 SLDs to a domain server. If one assumes that the location of the domain server also indicates the country the user lives in, it is possible to assign users with generic TLDs to countries. Clearly this method can be criticized since for example, a Spanish developer using an yahoo.com email account would be counted as a citizen of the USA. We return to this point later.

Time Zone (TZ) Another indicator is the time zone (TZ) indicated. A TZ like "EST" corresponds to several countries and therefore can not be used for our analysis. The same is true if 'time zone' has been left in its GMT default value. It is impossible to know which registrants ignored the TZ option and which ones ones ctually live in e.g. the U.K. Furthermore, the GMT corresponds to several countries. On the other hand, many TZs are unique and well defined and can be used to assign a country to users. For example, if one has chosen the TZ "Europe/Berlin", then this can be assigned to Germany. Here the underlying assumption is that users report their TZ correctly and that this indicates their usual place of residence.

Internet Protocol address (IP) If the Internet Protocol address (IP) of a user is available then this information can be used for geographic location using GeoIP. GeoIP is a technology that identifies geographic location of internet-connected devices based on their IP-range. The location of servers of internet service providers, universities etc., can easily be identified. Via these providers, the geographic location of internet users can be identified quite correctly. (The reader can try this out by visiting `www.maxmind.com/app/locate_my_ip`.) This technology lets us use the saved IP dates when this information is stored in the SRDA dumps. However, some IPs belong to ranges that

are assigned to regions instead of of specific countries and cannot be used for our purpose.

Identifying the geographical origin of OSS developers via ccTLD, IP and indicated TZ seems to be quite reliable, with IP providing the most accurate results. We check the overall reliability by pair-wise comparing the results that ccTLD, IP and indicated TZ deliver. For example to check ccTLD versus IP, we take the subset of users that have both an email address with a ccTLD and a saved IP listed in the data. We then count the number of times the both methods yield the same results. Similar cross-checking is performed for all methods. The results are presented in Table 2.1. As the reader

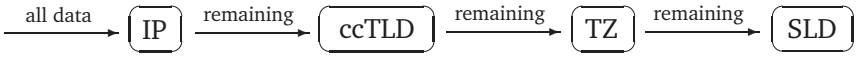
Table 2.1: Matching rates of the different identification methods

	IP	ccTLD	TZ	SLD
IP	100%	89.16%	87.29%	51.83%
ccTLD	89.16%	100%	80.45%	–
TZ	87.29%	80.45%	100%	56.45%
SLD	51.83%	–	56.45%	100%

can see, ccTLD and IP match 89.16% of the time, while IP and TZ match in 87.29% of all cases, and TZ and ccTLD match 80.45%. As previously mentioned we expect SLD to be the weakest method. Thus, not surprisingly, checking IP and TZ against SLD delivers matching rates of only 51.83%, and 56.45% respectively.

Based on this analysis we combine all four methods as follows (see also Figure 2.3): First, when possible, we identify users' geographical location

Figure 2.3: Process of geographical identification



via GeoIP. The remaining users are then identified via their ccTLD, if pos-

sible. The rest is then assigned to their country using the information about the TZ. The remaining 283,028 users are then assigned to a country based on their SLD information. Using this procedure, we are able to assign 1,315,263 users to specific countries (94% of our dataset). We also compare our results with and without using our weakest method (SLD). We find that the results do not materially differ.

As already mentioned, we are also interested in developers' activity levels. We use data about whether and how often user posted forum messages in 2006 as an indicator of activity. The SRDA contains information about the number of posted messages, stored in the Table 'forum'. This table is delivered by all the dumps from January 2006 until December 2006. The information in the columns 'msg_id' and 'posted_by' of the Table 'forum' link each user to his or her posted forum messages. We use this information to distinguish active developers (i.e. developers who posted at least once in 2006) from non-active ones. Our user data also allows us to count the total number of messages coming from each country. Tables 2.2 and 2.3 show the number of active developers and activity for the top 30 countries. We present the results both with and without our SLD identification. The number of active developers are highly correlated with the number of messages; the results with and without the SLD-identification are also highly correlated (both about 0.99).

Weighting all these data by population 2006 (World Bank, 2007), we get country specific data for the number of OSS developers per 1,000 inhabitants, the number of *active* OSS developers per 1,000 inhabitants, and the *level of OSS activity* (number of posted messages per 1,000 inhabitants). Because we have information about the activity of each individual developer, our data offers more information about global OSS activities than any previous non-survey study that we are aware of. The next section describes these results.

2.4 Results: The World-Wide Allocation of OSS Activities

In this section we present the results of our data mining and assignment analysis. We first look at the differences between active and non-active

Table 2.2: Active Developers, Top 30 Countries

without SLD			with SLD		
Rank	**Country**	**Active**	**Rank**	**Country**	**Active**
1	United States	85,485	1	United States	112,981
2	Germany	23,267	2	Germany	24,197
3	United Kingdom	13,031	3	United Kingdom	14,051
4	Canada	11,238	4	Canada	11,524
5	France	10,525	5	France	10,987
6	Australia	7,897	6	Australia	7,945
7	Netherlands	6,666	7	Netherlands	6,687
8	Italy	6,185	8	Italy	6,200
9	Spain	4,563	9	Spain	4,760
10	Sweden	4,546	10	Sweden	4,642
11	Brazil	4,028	11	India	4,163
12	India	3,824	12	Brazil	4,038
13	Russia	3,184	13	China	3,793
14	China	3,149	14	Russia	3,217
15	Belgium	3,026	15	Belgium	3,034
16	Switzerland	3,007	16	Switzerland	3,033
17	Austria	2,537	17	Austria	2,549
18	Poland	2,514	18	Poland	2,520
19	Denmark	2,314	19	Denmark	2,314
20	Hong Kong	1,861	20	Hong Kong	1,894
21	Norway	1,814	21	Norway	1,883
22	Finland	1,805	22	Finland	1,842
23	Singapore	1,685	23	Singapore	1,685
24	New Zealand	1,635	24	New Zealand	1,635
25	Israel	1,458	25	Israel	1,467
26	Argentina	1,456	26	Argentina	1,466
27	Czech Republic	1,443	27	Czech Republic	1,443
28	Mexico	1,401	28	Mexico	1,401
29	Japan	1,331	29	Japan	1,357
30	South Africa	1,211	30	South Africa	1,216

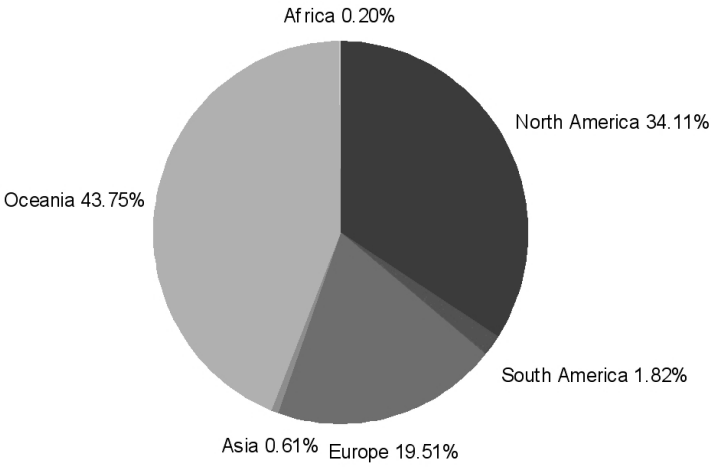
Table 2.3: Activity (Messages), Top 30 Countries

without SLD			with SLD		
Rank	**Country**	**SumMsg**	**Rank**	**Country**	**SumMsg**
1	United States	7,734,231	1	United States	8,842,906
2	Germany	1,807,233	2	Germany	1,839,842
3	United Kingdom	1,238,922	3	United Kingdom	1,282,156
4	Canada	1,064,001	4	Canada	1,078,637
5	France	826,659	5	France	845,302
6	Australia	720,326	6	Australia	721,693
7	Netherlands	570,732	7	Netherlands	571,198
8	Italy	433,354	8	Italy	433,793
9	Sweden	361,550	9	Sweden	367,379
10	Spain	329,655	10	Spain	340,567
11	Belgium	260,754	11	Belgium	261,111
12	Switzerland	244,784	12	Switzerland	245,485
13	Russia	238,709	13	Russia	239,824
14	Austria	220,552	14	Austria	220,814
15	Brazil	205,466	15	India	211,735
16	Denmark	198,132	16	Brazil	206,042
17	India	197,009	17	Denmark	198,132
18	China	160,279	18	China	184,113
19	Japan	148,305	19	Japan	149,500
20	Norway	136,902	20	Norway	140,096
21	Poland	136,555	21	Poland	136,798
22	Finland	132,489	22	Finland	134,024
23	New Zealand	119,515	23	New Zealand	119,515
24	Hong Kong	116,515	24	Hong Kong	117,749
25	Argentina	116,492	25	Argentina	117,419
26	Czech Republic	114,713	26	Czech Republic	114,713
27	Romania	111,843	27	Romania	111,843
28	Singapore	106,009	28	Singapore	106,009
29	Israel	98,492	29	Israel	98,653
30	Mexico	81,956	30	Mexico	81,956

developers. On average only 19.88% of all registered developers were active in 2006 (first, second and third quantile shares of active developers are 12.5%, 18.68%, and 23.53% respectively). These facts support the idea that being a registered OSS developer is not the same thing as being an active developer. Clearly it is more interesting to know where the active developers live. In addition, focusinging on the number of developers including the inactive ones could be misleading if such data is used for country-specific research, policy advice, or for comparative studies analyzing the impact of country-specific factors on OSS. We therefore focus on OSS *activities* and *active developers* in what follows.

First we look at the share of activities that come from different regions. Figure 2.4 shows the activity level per capita for six world regions. The

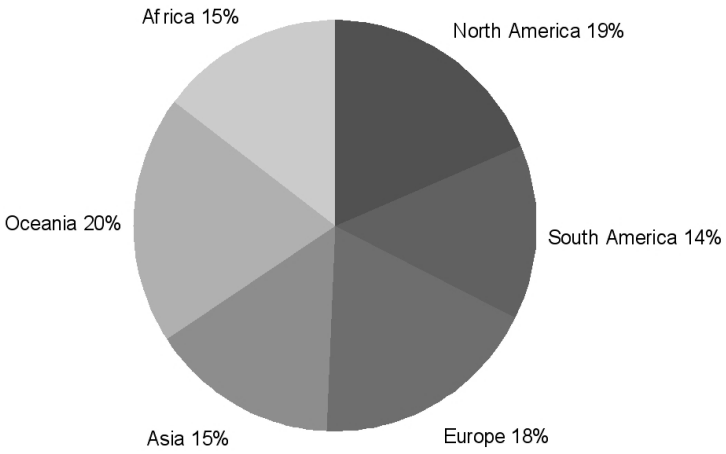
Figure 2.4: Activity Level (Messages per Inhabitants)



distribution of active developers per capita is not materially different. The reason is that the average activity level by active developers of regions is remarkably similar (15-20%), see Figure 2.5. Despite very different distributions of active developers per capita and activity levels across regions,

the average developer posts roughly the same number of messages all over the world.

Figure 2.5: Average Activity of Active Developers



The remarkable similarity of the average active developer's activity also holds at the level of countries. To see this, compare Figure 2.6 and Figure 2.7. The two figures depict the worldwide allocation of active OSS developers and of the OSS activity-levels: the two maps look similar.

Figures 2.6 and 2.7 also show how OSS activities differ over the world. Some countries display high degrees of activity and large numbers of active developers per capita. However, many other countries show virtually no active OSS developers. Figure 2.8 illustrates this unequal distribution with a quantile plot of activity levels per countries.

OSS also seems to be a phenomenon of the developed world: in 2006 85% of all active developers live in one of the OECD countries and posted 88% of all messages. One might guess that OSS is a rich countries' phenomenon, i.e. is correlated with GDP. To explore this hypothesis, we divide the number of active developers by each country's per capita GDP (in purchasing power parity) for 2006, data source is World Bank (2007). Figure

Figure 2.6: World Map of Active OSS Developers

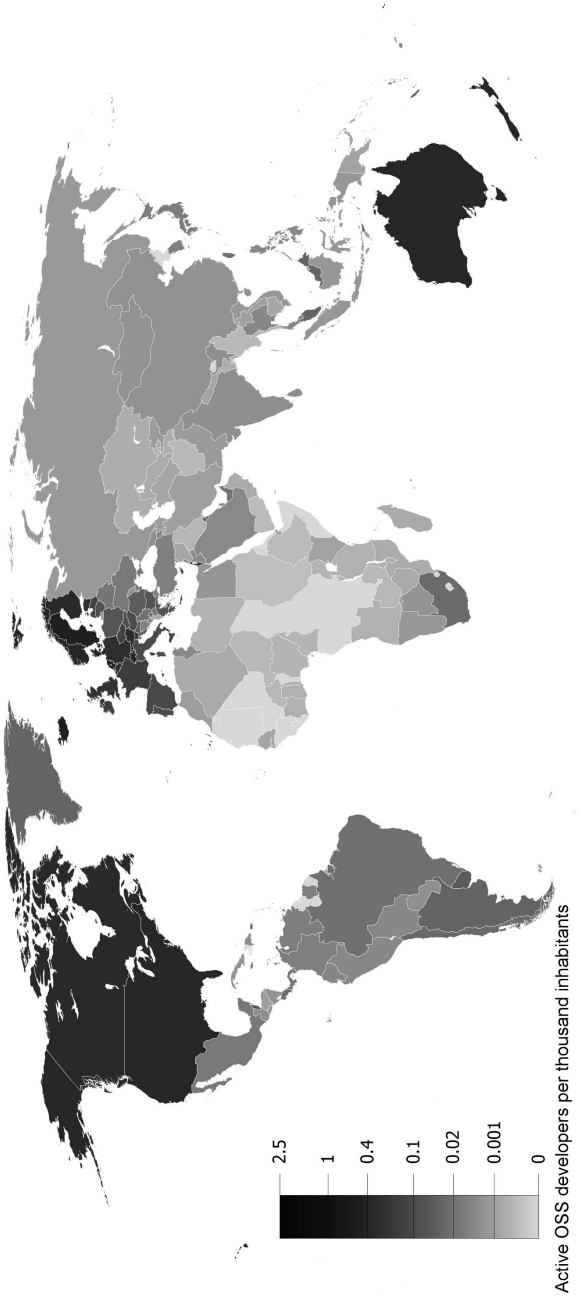


Figure 2.7: World Map of OSS Activity-Levels

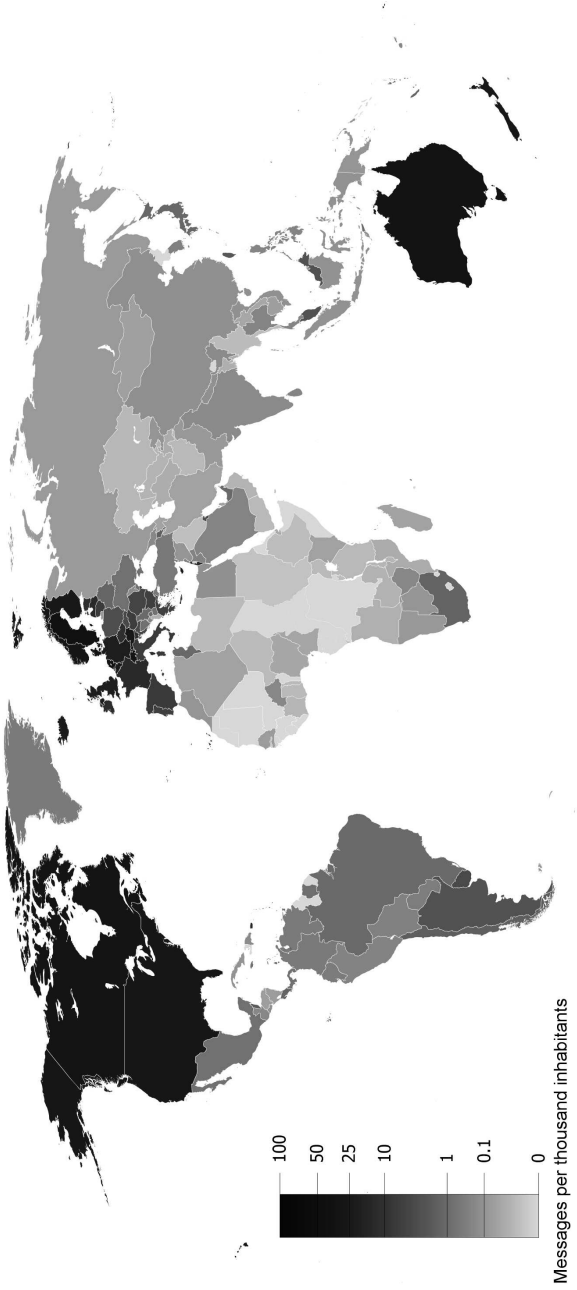
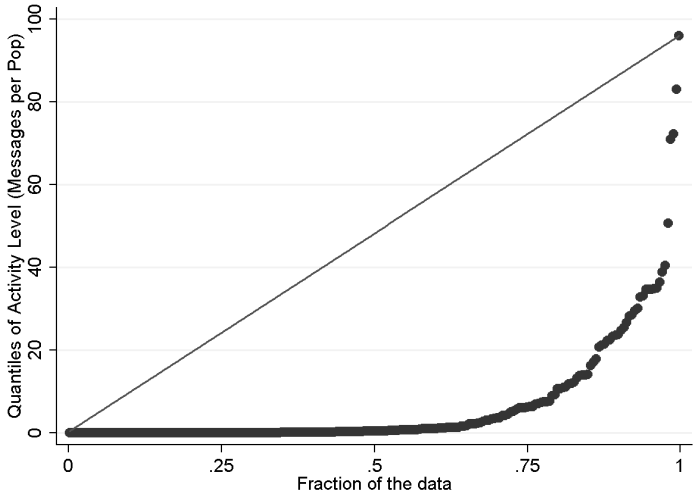


Figure 2.8: Distribution of Activity Levels (Quantile Plot)



2.9 shows the results.[3] Still the allocation is very unequal. Thus, the phenomenon of OSS cannot be explained solely by GDP per capita.

We also analyze the impact of the internet. Internet access is obviously a precondition for OSS in general since all OSS interactions take place on the internet. Indeed, there is no way to become a registered SourceForge developer without it. We use data from the International Telecommunication Union (2006) on the number of internet users, i.e. about the 'internet-population' of each country. Figure 2.10 shows the number of active developers per internet user[4] (the results for activity per internet user are quite similar). This shows that adjusting for internet access leads to more even distributions than adjusting for GDP or the unadjusted raw distributions. Nevertheless, there remain differences. Thus, internet usage alone cannot explain differences in world-wide OSS activities.

[3] The patterned areas are countries whose GDP per capita in 2006 is not available.

[4] The patterned areas are those countries with lack of data regarding the number of internet users.

Figure 2.9: World Map of Active OSS Developers weighted by GDP per capita

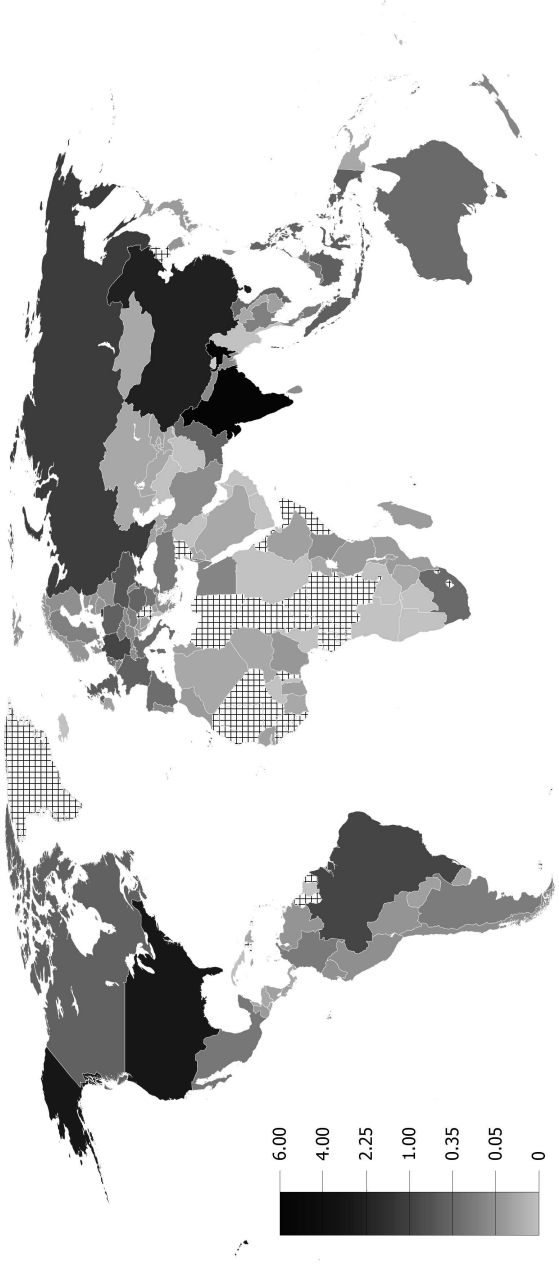
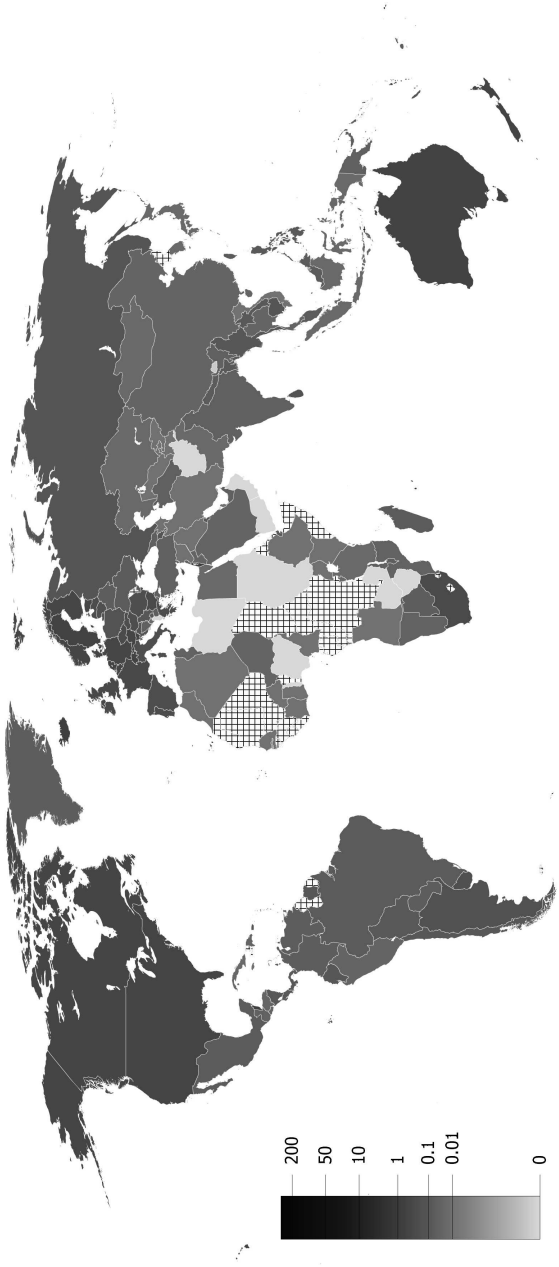


Figure 2.10: World Map of Active OSS Developers per Thousand Internet Users



2.5 Summary and Outlook

Software industry data typically ignore OSS activities. Reliable data that distinguish between registered OSS developers, active OSS developers, and OSS activity level can help to correct this. This chapter presents a more complete picture. This will be useful for both, policy makers and businesses. Our country data will also be useful for future cross-country studies and research on the supply of OSS.

Analyzing IP address, email and time-zone from the SourceForge Research Data Archive allows us to geographically identify 94% of all registered OSS developers in 2006. The information about the number of posted messages provides a good proxy for activity of each developer. Based on this we analyze the world-wide distribution of OSS activities. Geographic origin seems to matter. The allocation of active OSS developers (and of OSS activities) is geographically uneven. This still holds if one corrects for population or per capita GDP. Adjusting for 'internet-population'—i.e. the number of internet users—makes some but not all of these differences disappear.

Given that the worldwide distribution of OSS activities cannot be completely explained by GDP or number of internet users, the question arises which further factors have an impact on OSS. Cultural and institutional factors are potential candidates. Analyzing this should also help to get a better understanding of what OSS is about. Our country-specific data on the number of OSS developers, the number of active OSS developers, and the level of OSS activity provide a good foundation for such research. In the next chapter we undertake such an analysis and examine the impact of country-specific cultural and institutional factors on the number developers, active developers and activity.

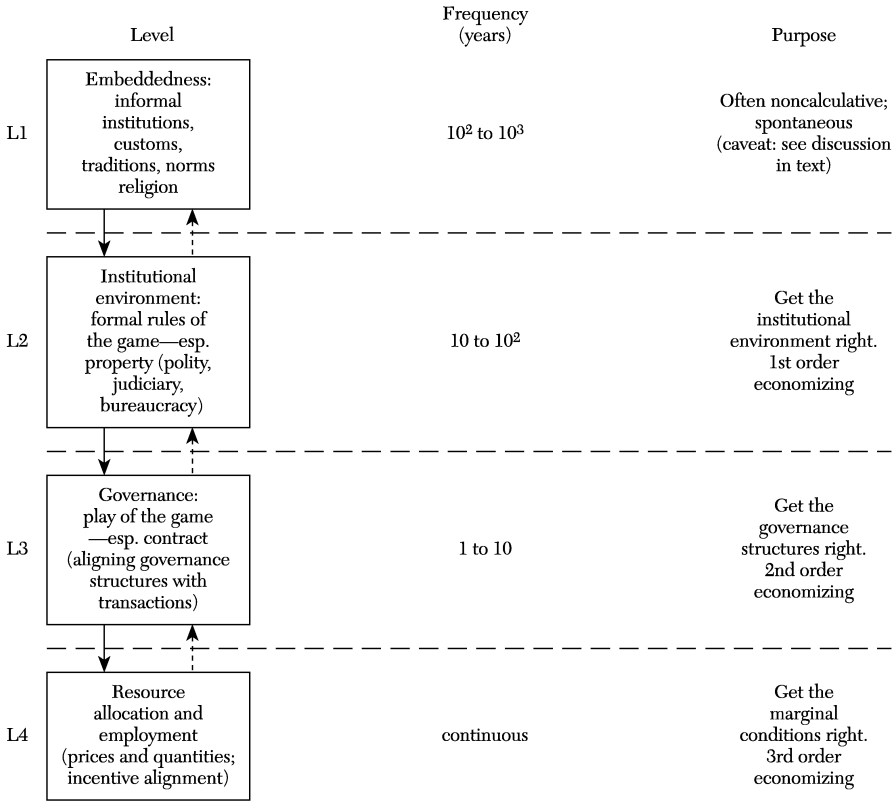
3 Institutions, Culture, and Open Source

3.1 Introduction*

This chapter analyzes the influence of a country's institutional and cultural framework on the supply side of OSS. In this context we refer to Williamson's analytical framework (Williamson, 2000, see Figure 3.1) and point out that, so far, economic research on OSS focus on level three and four only. As the overview in Section 1.3 shows, the existing literature on the economics of OSS deals with the intrinsic and extrinsic motives of the OSS developers, the micro-level institutions like governance structures and licenses, the the impact of OSS on competition and market outcome, and the incentives and role of firms. Thus, with respect to OSS, there is still lack of knowledge regarding the levels one and two, i.e. regarding the so-called "embeddedness" (informal institutions, customs, traditions, norms, religion) and the institutional environment (formal rules of the game, in particular property rights). This study fills this gap, as we are interested in the conditions for OSS activities on the level of society. We take into consideration the microeconomics of OSS and search for the conditions for these aspects. For this purpose, we perform a cross-country study analyzing how the per capita number of a country's OSS developers registered at SourceForge is shaped by institutional and cultural factors. In particular, we take into account aspects of the legal system and regulation, social capital, the openness to novelty, the degree of individualism/self-determination of a society as well as its attitude toward competition, with some of these measures based on principal component analysis.

*This Chapter is based on von Engelhardt and Freytag (2010)

Figure 3.1: Williamson's four interrelated levels of social and institutional analysis



(Source: Williamson, 2000, p 597)

The remainder of the chapter is structured as follows: In Section 3.2, we discuss the theoretical foundations and derive the hypotheses for the empirical study. In Section 3.3, we operationalize the variables and describe the data and its sources. This data is used to perform the empirical assessment in Section 3.4, where the regression results are presented. In Section 3.5 we compare and discuss the results before we end with a summary in Section 3.6.

3.2 Theoretical Considerations and Hypotheses

As shown in the last chapter, countries differ in the number of OSS developers per capita as well as in the level of OSS activity. These differences cannot be solely explained by GDP or access to the internet.

In general, cultural and institutional factors that belong to level one and two of Williamson's framework shape human interaction and therefore have an impact on the microeconomic level. Hence, in order to derive hypotheses about the influence of institutional and cultural factors on OSS developers and their activities, we will link insights about the microeconomics of OSS with the level of institutional and cultural factors.

This is a relatively new approach. The only study (we are aware of) linking cultural factors with the geographics of OSS developers is Ramanujam (2007).[1] Ramanujam uses data from Ghosh (2006) and Hofstede's cultural indicators to analyze how differences in national culture affect or influence the participation in OSS. He links the geographical distribution of developers with the four dimensions of national cultures considered by Hofstede (1991). Ramanujam states a positive correlation between the share of OSS developers and 'Individualism', whereas 'Power Distance' and 'Uncertainty Avoidance' are negatively correlated each. However, the results should be interpreted with care, as there is no control for aspects like number of inhabitants, GDP, internet access, etc. Furthermore, with respect to OSS

[1] Ramanujam's hypothesis is that "Cultural differences amongst the programmers from different regions lead to measurable differences in their participation in the open source movement. In other words national cultural differences influence the participation of programmers in development of OSS" (Ramanujam, 2007, p 16). When interpreting his results he gives some plausible explanations for his findings. Nevertheless, study seems a bit vague with respect to theoretical foundation.

contribution Ramanujam (2007) distinguishes only four regions, whereas this study runs regressions with data from about 70 countries, and analyzes several cultural and institutional factors including norms and attitudes.

3.2.1 The Role of Culture, Informal and Formal Institutions

Institutions define the incentive structure of a society and are therefore the underlying determinants of economic performance. Human interaction is structured and shaped by formal institutions (e.g., rules, laws, constitutions) as well as by informal institutions (e.g., norms of behavior, conventions, self-imposed codes of conduct). (North, 1994, p 359 f). According to North, informal institutions belong to "the heritage called culture" (North, 1990, p 37). This is in line with Williamson's framework, as his level 1 ("ebeddedness") is characterized by the set of informal institutions, namely customs, traditions, norms and religion (Williamson, 2000, see also 3.1). Therefore some economists analyze culture in terms of informal institutions like social conventions or individual beliefs like interpersonal trust or (rational) cultural beliefs that are self-enforcing (Guiso et al., 2008; Greif, 1994, p 915; Myerson, 1991).

However, although informal institutions belong to the sphere of culture, not everything belonging to culture is an (informal) institution. Although culture shapes human interaction, some parts of culture are not institutions by definition, as they lack of enforcement characteristics. Nevertheless, this part of culture also affects economic behavior, as it is linked to individual values and preferences. Research focusing on this aspect of culture can be found in e.g. Rabin (1993) and (Akerlof and Kranton, 2000, using the concept of 'identity'). Bowles (1998) treats preferences as cultural traits, and Bisin and Verdier (2001) model intergenerational cultural transmission as transmission of preferences, while Fernández and Fogli (2009) analyze the impact of culture in terms of preferences and beliefs on women's work and fertility.

In addition, the different aspects and dimensions of culture can influence each other, and there are also interdependencies between the sphere of culture and formal institutions. How culture influences the implementation of formal rules was for example analyzed by Greif's seminal article about the impact of cultural beliefs on the introduction of different organizations

(Greif, 1994). Other examples for research dealing with the interaction of culture and (formal) institutions are Tabellini (2007, 2008a), the research dealing with informal vs. formal institutions and the transition of economics.

This discussion of the literature[2] leads to two implications regarding the role of culture:

1. Culture has an impact on economics, as it influences economic behavior either in forms of social conventions, in forms of beliefs, or in forms of individual values and preferences.

2. Culture "embeds" and shapes lower-level institutions. This means that certain cultural characteristics can foster or hinder the implementation and/or functioning of institutions.

3.2.2 The Phenomenon of OSS and Institutional and Cultural Factors

Being a social-economic phenomenon, OSS development has several dimensions, which are interconnected and can overlap. First, OSS has similarities to technical science and scientific culture: Its principle of openness and reputation mechanism remind of open science. Many OSS developers are at universities. Historically the idea of OSS comes from software-

[2]Additionaly to the literature mentioned in the text there is a variety of research. The following list of further examples is not intended to be exhaustive, but rather to give an impression of the range of research on culture and economics. An overview and an introduction can be found in Fernández (2008), see also Guiso et al. (2006) for a summary of research. A discussion about the concept of culture in economic research is Heydemann (2008) vs. Nugent (2008), while Jackson (1993) discusses "culture, society and economic theory". Scholars like Svetozar Pejovich and Eckehard F. Rosenbaum have analyzed the role of culture in the process of transition of (former) socialist economies Pejovich (2003); Rosenbaum (2001). This is related to the interplay of on formal and informal institutions like e.g. Williamson (2009) deals with. Tabellini (2008b) analyzes norms and values on cooperation, taking into account (a) that individuals also value the act of cooperating per se, and (b) the social embeddednes of the players ("within a circle of socially connected individuals"). Henrich (2000) analyzes the impact of culture on ultimatum game bargaining, and Alverson (1986) examines games that "play people". And Bénabou and Tirole (2006) connect beliefs and voting in a model with an "American" and an "European" equilibrium in endogenously shared ideology.

science and is rooted in scientific culture, implying the willingness to help each other and to discuss problems in online-forums. Additionally, (open source) software development is connected to the search for new solutions. And OSS has a strong technical aspect: software can be seen as a logical machine, and is clearly connected to computers and the internet. Second, OSS is a public good game, or collective action, being linked to complementary assets on the individual level; in that sense, contribution to OSS is always a means to an end. Either somebody (further) develops the code for own purpose or receives utility from doing so. In the latter case OSS is a (globally coordinated) leisure activity, a task that is done for self-fulfillment and self-determination. There are also extrinsic motives such as building reputation signals or generating income with OSS business models, i.e. selling complementary products like service or hardware. Third, OSS is a new intellectual property right paradigm. The existence and success of OSS seem to challenge the conventional wisdom about the proper role of intellectual property rights. It is important to notice that the several OSS licenses (and the OSS business models) rely on copyright law, and that the governance of the different OSS projects is to some extend based on trademarks etc. Another dimension of OSS is its entrepreneurial spirit: Beside the fact that there exist a variety of OSS business models, OSS projects are set up or supported by individuals who want to solve a problem or implement a new feature. Thus OSS in general is based on the idea of individual initiative. Clearly the openness of the code is a precondition here but a sense for pragmatic solutions helps also.

For these aspects we develop hypotheses of how institutional or cultural factors have an influence on OSS development. We break up the phenomenon OSS into several elements, identify the more general, underlying aspects, and then connect these with the institutional or cultural factor that is or is not in favor of the particular aspect. It is important to notice that these general aspects are not exclusively linked to OSS. For example: We argue that OSS is an example of an individualistic, self-deterministic behavior. In a society with a strong culture of individualism/self-determination, we expect more individualistic, self-deterministic behavior. As OSS development is such behavior, we expect to see more OSS. We thus argue that it is the cultural spirit of a country that makes it more likely that individuals choose certain tasks with specific characteristics, in our case develop OSS.

3.2.3 The Hypotheses

This section derives the hypotheses. Before we discuss the relation of the different institutional and cultural factors to OSS development in detail, we first take into account the information and communications technologies (ICT) because of two reasons: First, the supply- and demand-side of the ICT-sector are of importance for OSS. One extrinsic motive to develop OSS is to build up reputation signals for the job market, and the size of such jobs markets is linked to the supply side of ICT. Furthermore, some OSS activities are connected to OSS business models. Here the potential size of the demand-side is of importance. Second, internet access is a technical and cultural precondition to participate in OSS. Without internet there is no access to the online community of OSS developers. In addition, without some internet experience, there are mental barriers to join the OSS community which is rooted in the cyber-space culture ("hacker-culture"). Taking this together our first hypothesis is:

H1: ICT is beneficial both for the number of OSS developers and the OSS activity level.

Next we discuss the cultural and institutional factors. We first focus on science: OSS development is a collaborative way of developing novelty. The process of (open source) software development is a search for new solutions, i.e. an innovative process as such. Thus, OSS development can be described as "coordinating innovation" Kugler (2005). Additionally, the rise of OSS is an innovation at the level of how to organize software development, and some authors discuss it as a new intellectual property paradigm (Maurer and Scotchmer, 2006) or regard OSS as an "intellectual property revolution" (Pisano, 2006).

In this context we make use of the notion of culture that connects culture with preferences and values: In societies that are more open to novelty, in particular that are more open to new ideas, a higher share of people would prefer new ideas. Such preferences are a good precondition for the adoption of the OSS model of software development and also for active participation, i.e. the search for new solutions. Therefore we expect the following:

H2: A preference for new ideas on the level of society has a positive impact on the number of OSS developers as well as on the OSS activity level.

Another aspect is also related to science, i.e. the similarities to technical science and scientific culture: OSS itself has a strong technical aspect. It is a novelty from the 'cyber space', clearly connected to computers and the internet and therefore to the technical aspect of scientific progress. Additionally, software development is an art to build a logical machine. Second, historically the idea of OSS stems from software-science and is therefore rooted in scientific culture. For example, the best known OSS license, the GPL, was developed by Richard Stallman. Richard Stallman worked in the MIT Artificial Intelligence Lab. When in the 1980s more and more software became CSS Stallman started the GNU project in order to defend and foster a 'free'—in terms of 'open'—culture of software development. This finally led to the GPL licenses (see Section 1.2.1). Still today, the OSS community has scientific-alike aspects: There is a culture of discussing problems and helping each other in online-forums, and many OSS developers are at universities. As for openness and the reputation mechanism, similarities to open science can be observed (Dalle and David, 2005; Lerner and Tirole, 2002; Giuri et al., 2002).

We now argue that a positive attitude towards technical science and/or scientific progress at all is in favor of OSS. Such a pro-science culture (in terms of attitudes, i.e. preferences and values) may support the science-alike formal and informal institutions of the OSS community. Based on this, we state the following hypothesis:

H3: A positive attitude towards scientific progress has a positive impact on the number of OSS developers and on the level of OSS activities.

Contributing to OSS can be a means to an end, because the developers directly receive utility from doing so. For example, Hars and Ou (2002) found that "self-determination" was with about 80% agreement the strongest intrinsic motive. Other authors report that "fun" and enjoyment of programming work itself or of solving problems, and an intellectual challenge are important motives for individuals to contribute to OSS (Lakhani and Wolf, 2005; Hertel et al., 2003; Lakhani et al., 2002). Thus OSS seems to

meet values and preferences which are connected to a culture of individualism and self-determination. Here we have again the link between culture and preferences: in a culture with a higher degree of individualism/self-determination, one would expect that more people engage in individualistic hobbies. As research on the intrinsic motives of OSS developers suggest that OSS development is an activity that fits such preferences, we expect to see an impact on OSS development:

H4: The degree of individualism/self-determination of a society has a positive impact on the number of OSS developers as well as on the OSS activity level.

Although individual OSS contribution is a means to an end, linked to (complementary) assets (i.e. intrinsic or extrinsic motives), the OSS development process still is a public good game. This brings us to social capital, which is related to ties between people. While some refer to the number of ties only, others stress the features, strength or quality of such ties, which includes aspects like trust. Probably the most known (and widely accepted) definition of "social capital" is by Putnam (1993b,a). Putnam states that social capital "refers to features of social organization such as networks, norms, and social trust that facilitate coordination and cooperation for mutual benefit" (Putnam, 1993a, p 67). Therefore measures of social capital can take aspects like number of people somebody is (weakly) connected to, but also aspects like social, i.e. interpersonal trust and social engagement into account. We focus on interpersonal trust in this study.

Thus, the theoretical concept (or aspect) of culture we refer to in this context is culture as beliefs, namely interpersonal trust.[3] For our purpose it is not relevant where this country-specific level of interpersonal trust comes from (e.g. as result of a general 'social' game, or from several sub-games): What matters is the implication of (ex ante) interpersonal trust on the behavior of individuals with respect to the international public good game

[3] In economics the notion of 'social capital' differs. While some refer to the number of ties only, others stress the features, strength or quality of such ties, which includes aspects like trust. Therefore measures of social capital used in the literature take into account aspects like number of people somebody is (weakly) connected to, or social, i.e. interpersonal trust and social engagement. We focus on interpersonal trust in this study.

OSS. First, individuals with more trust will expect less free-rider behavior and more reciprocal behavior. Thus, they are more likely to contribute to OSS themselves. Second, the literature on public good problems indicates that interpersonal trust has a positive impact on cooperation and reciprocate behavior (Yamagishi et al., 2005; de Cremer, 1999; Ostrom, 1998; Yamagishi and Yamagishi, 1994). Hence having more individuals with higher interpersonal trust should yield more OSS contributions, more reciprocate behavior etc. from that specific country. Such behavior is then stabilized: the ex ante beliefs are supported by the outcome of the game as OSS is a successful public good game. Voluntary code-contribution and reciprocity is part of the OSS community culture (Ghosh et al., 2002b; Lakhani et al., 2002). Therefore we expect the following:

H5: Social Capital in terms of interpersonal trust has a positive impact on the number of OSS developers as well as on the OSS activity level.

So far, we have concentrated on intrinsic aspects (motives, preferences beliefs). Now we turn to extrinsic motives and incentives. In particular we will focus on aspects of OSS that are linked to the enforcement—the enforcement mechanisms respectively—of particular institutions. We first analyze an informal institution of OSS (the reputation mechanism) before we discuss the impact of the protection of intellectual property rights.

Extrinsic motives of OSS developers are for example self-marketing, peer recognition and reputation within the community (Hars and Ou, 2002; Lakhani et al., 2002). Career aspects are directly linked to extrinsic motives like the improvement of programming skills, i.e. the investment in human capital, and the aim to build up reputation signals for the job market (Lakhani and Wolf, 2005; Hertel et al., 2003; Ghosh et al., 2002b; Hars and Ou, 2002; Lakhani et al., 2002, in all cases these motive were stronger than the motives related to peer recognition). We sum up these two sets of motives as “reputation mechanism”. Such a reputation mechanism is an incentive structure based on the merit principle. In addition, the relevance of such performance signals indicates competition, especially when it comes to the job-market.

The reputation mechanism is an informal institution of the OSS community that has a peer-based (positive) enforcement mechanism. This enforcement mechanism itself is of interest with respect to the role of culture:

Positive attitudes towards the merit principle and competition foster and support such enforcement mechanisms of performance based reputation. If more individuals will accept the idea of individual performance signals, more peers will be willing to reward good contributions. Finally, more agents will see the need for and have a preference for the achievement of such signals. In a nutshell, we analyze how culture in terms of preferences and values can foster or hinder the functioning of an institution, namely the effectiveness of its enforcement mechanism. Thus, in a country with a more positive attitude towards competition and the merit principle, it is more likely to find software developers or students who engage in OSS with the goal to send reputation signals (for "sportive" peer-competition as well as for career aspects).

H6: A culture of positive attitudes toward competition and the merit principle has a positive impact on the number of OSS developers as well as on the OSS activity level.

We now turn to the enforcement of a formal institution, namely the de facto protection of intellectual property rights (IPRs). OSS challenges the traditional wisdom of the exclusive use of IPR, and is often seen as a new IPR paradigm. So, at a first glance, the relationship between protection of IPRs and OSS may not be clear. Some parts of the OSS movement, like the GPL-founder Richard Stallman, argue against intellectual property while others oppose this.[4] However, in fact OSS licenses are real legal licenses, since they define the scope of transferred rights and are based on copyright law de Laat (2005). Especially Stallman's GPL uses a so-called 'Copyleft'-principle which ensures that the licensed software stays "open". Basically

[4]Some parts of the open source community argue in an anti intellectual property way. The "Free Software Foundation" opposes the use of the term "intellectual property", and its president Richard Stallman refuses the idea of intellectual property, arguing that because of moral reasons no one should be allowed to claim property rights on information or knowledge. This view is opposed by figure like Eric S. Raymond, co-founder of the Open Source Initiative. Raymond supports the idea of property right claims, and hence also of intellectual property rights, but simply argues that proprietary software (in the sense of CSS) is simply an inefficient way of developing software (see Weber, 2004a). Others like e.g. Greg Perkins also point out that "Open Source depends on the idea of the individual human right to private property" Perkins (1998).

this is achieved via restricting the right to redistribute in the following way: Any further developed software as well as any derived work must be licensed as a whole under the GPL. Thus, intellectual property law is used to ensure that OSS stays OSS as the GPL is based on copyright (Gehring, 2006, pp 62, 70). In addition there exist a variety of different OSS licenses differing in how they restrict the usage of the code. In particular firms 'owning' OSS projects make use of sophisticated licenses and dual-license strategies, as it is crucial for them to define exactly what is exclusively owned and what not.[5] Obviously, such legal arrangements are only possible and effective if intellectual property rights are respected and such licenses can be enforced.[6] Furthermore, the OSS incentive and governance structures are based on trademarks. Core developers of an OSS project control the project by using passive control rights that are their exclusive rights to decide whether to accept or reject contributions (see Section 4.3.3.2 in the next chapter, see also McGowan, 2001; Wendel de Joode et al., 2003, p 20). These passive control rights are enforced by using the concept of ownership regarding the database in which the software is stored and the name—thus, the trademark—of the project. This prevents cloning of projects and supports the signaling function of the project's name. Here protection of IPRs supports indirectly the OSS governance structures and the informal institution 'reputation'. Thus both, the non-commercial part of the OSS community as well as the firms involved, benefit in practice from the possibility to define and enforce IPRs. Therefore we state the following:

H7: The protection of intellectual property rights has a positive impact on the number of OSS developers or on OSS activity level.

Finally we take into account a set of formal institutions: the degree of regulation of economic activity and its impact on OSS development. We argue that more regulation is not in favor of OSS development and even hinders it. In general, a high degree of regulation increases (transaction) costs of entrepreneurial activities and individual initiatives. This depresses such

[5] For more details the different licenses see e.g. Lerner and Tirole (2005); for dual licenses see Välimäki (2003).

[6] See e.g. Kumar (2006) on the GPL, for current examples of "the GPL in court" visit `http://gpl-violations.org/`.

activities, and thus also OSS. As argued above, OSS has an entrepreneurial spirit as it is based on the idea of individual initiatives and pragmatic problem solving. The opportunity to have access to and to flexibly use the code (individualize, further develop, etc.), fosters additional entrepreneurial activities. Hence OSS can be a precondition for OSS based start-ups, enable firms to run OSS business models etc. Strong and distortive regulation of economic activities in a country has a negative impact on doing business and thus also on OSS by firms. Furthermore, in the long run strong regulations can also affect the attitudes of the inhabitants negatively: It should not only decrease the number of such activities but also the entrepreneurial spirit in general. So the theoretic argument is that entrepreneurial attitudes or spirit foster entrepreneurial activities. The "payoff" of such activities depends on regulation. Thus high regulation increases transaction costs and hence lowers the payoff. This finally leads to less entrepreneurial activities and in the long run also to less entrepreneurial spirit. Because of its characteristics, OSS belongs to entrepreneurial activities, is based on entrepreneurial spirit respectively. Thus we should see a negative impact here:

H8: A high degree of intense economic regulation has a negative impact on the number of OSS developers as well as on the OSS activity level.

In addition, some control variables are necessary. The data about the geographic origin of OSS developers show that most OSS contributions come from developed countries. Therefore, we control for GDP per capita. Furthermore, we control for education, because studies like Ghosh et al. (2002b) indicate that OSS developers are well-educated software engineers or ICT students.

3.3 The Data

For information about the geographical origin of OSS developers registered at SourceForge we make use of the same dataset as in chapter 2. Regarding the institutional and cultural aspects we make use of date available from different resources:

3.3.1 Data on GDP, Education and ICT

The probability that a country's inhabitant becomes an OSS developer rises with the degree of economic and technical development, in the latter case mainly with the access to the internet. We take into account the GDP ppp (purchasing power parity) per capita for 2006. As a measure for education we use the combined gross enrollment ratio for primary, secondary and tertiary schools with a four-year lag. We took both from Norris (2009), the original data source are World Bank (2007) and UNDP (2004). We have to take into account aspects of ICT. However, worldwide data about e.g. the number of software developers, size of the software sector or other differentiated data about the ICT sector are poor. The best data available refer to internet access.[7] Thus we use the number of internet users per inhabitants ("inet users") as a proxy here. The data for this come from the International Telecommunication Union (2006).

3.3.2 Data on Cultural Factors and Social Capital

One main source for our analysis is the World Values Survey (WVS) offering a wide range of country-specific cultural data. It is often used in cross-cultural research. We refer to this for our cultural variables about using data from the waves of 1990, 1995/1998 and 1999/2000 from the online-dataset at `www.worldvaluessurvey.org` (category `Online Data Analysis').

However, not all questions were asked in all countries, and additionally not in all interviews. Thus we have to correct for that and eliminate all those with too little overall coverage.[8] Some of the questions have an

[7] At least for some countries, data about the share of employees working in the ICT sector is available. But as internet access is a precondition for OSS, we want to have this in our regression in any case. But with share of ICT employees and internet users we clearly run into problems of multicollinearity here: each internet access must have been installed by someone working in the ICT sector. Therefore we decided to leave this data out and use internet usage only. We also decided not to use real prices of ICT because of a lack of data.

[8] In case of questions that were not asked in 100% of all interviews in a country (but with a sufficient high percentage) we additionally have to correct the percentage of answers, as the numbers one receives from the WVS online-dataset always sum up with the "not asked"- share to 100%.

ordinal scale for the possible answer. Although it is very common to use the mean of such answers, this is quite critical from a methodological point of view, as in this case one treats ordinal scaled data like being on an interval scale. It is better to choose a certain threshold, i.e. for example count the percentage of answers with scale 4 or smaller. As we want to be able to distinguish groups (here countries) from each other with respect to a certain characteristics, a good way to find such a threshold is to "ask the data". Thus, we look at the direction the answers point to, choose those of interest, and then set different plausible thresholds. In a next step we compare the variance, and choose those with the higher variance, as this is an indication by the data that we made the right cut in order to measure the difference of the respective category. (If variances were close to each other, we choose that threshold with the distribution closer to the normal distribution). Whenever we refer to WVS data henceforth, they were, if necessary, treated in the way just described.

3.3.2.1 Social Capital: Interpersonal Trust

Interpersonal trust is measured by the average percentage of respondents saying 'most people can be trusted'. (The question is "Generally speaking, would you say that most people can be trusted or that you need to be very careful in dealing with people?" Possible answers are "Most people can be trusted" and "Can't be too careful"). We label this "interpersonal trust".

3.3.2.2 Degree of Individualism/Self-Determination

To measure the degree of individualism/self-determination is more complicated. It is based on Hofstede's definition of individualism, which "pertains to societies in which the ties between individuals are loose: everyone is expected to look after himself or herself and his or her immediate family" (Hofstede, 1991, p 52). He developed the individualism index for 50 countries based on a world-wide survey of IBM employees that was carried out during 1978-83. The questions the individualism index was built upon was whether the job leaves sufficient time for personal and family life, considerable freedom to adopt own approaches, includes challenging work, offers opportunities to improve and learn new skills, etc. (Hofstede, 1991, p 49

ff). Hence, based on these categories, high scores in individualism indicate the prevalence of individual interest in a society, i.e. in a sense that people would like to (and can) "do what they want to do". We use an updated and further developed version of Hofstede's measure, namely a merge of ratings provided by Triandis' and Hofstede's scores (Suh et al., 1998, p 485; Diener et al., 2000; Oishi, 2000).

Obviously the concept of 'individualism' in the tradition of Hofstede's definition should correlate with leisure time preferences, preferences for independence and self responsibility etc. Here we can again make use of the WVS data. Treating the data as described above lead to the following categories for "leisure time" and "self-responsibility":

- Leisure time: % of all respondents of a country saying "1 Very important". (Question asks how important leisure time is in ones life.)
- Self-responsibility: % of all respondents of a country ranging from 1 to 4. (Question asks to put oneself on a range 1 to 10 expression own opinion, with 1 = People should take more responsibility to provide for themselves, vs. 10 = The government should take more responsibility to ensure that everyone is provided for.)

Additionally, the WVS data delivers the percentage of all respondents of a country who mentioned that "feeling of responsibility" is an important quality children shout learn at home. (They were given a list of qualities that children can be encouraged to learn at home. They should choose up to five they consider to be especially important.).

We want to have one single measure for the degree of individualism/self-determination. Assume that a certain characteristic (like the degree of individualism/self-determination) cannot be measured directly, but several indicators for this characteristic are available. Then principal component analysis (PCA) is a useful tool to identify the meaningful underlying variable(s) and construct this based on the data available. In other words, the PCA tries to find components that explain the maximum amount of variance, reducing the dimension of the data and detecting the structure in the relationships between variables. We construct a principal component labeled "self-determ/indiv", that consists of the individualism scores mentioned above, "leisure time", "self-responsibility" and whether a child

should learn responsibility. The Kaiser-Meyer-Olkin measure of sampling adequacy of "self-determ/indiv" is 0.683.

3.3.2.3 Attitudes Toward Competition and the Merit Principle

For the degree of positive attitudes toward competition and/or the merit principle we are also able to construct a principal component, labeled "competition/merit" consisting of variables from the WVS, measuring attitudes towards income differences as incentives, whether competition is perceived as good or harmful, and the importance to teach a child independence:

- Income differences as incentives: % of respondents of a country ranging from 7 to 10 (Question asks to express own opinion, using a range from 1 = Incomes should be made more equal, up to 10 = We need larger income differences as incentives.)
- Competition is good: % of respondents of a country ranging from 1 to 2. (Question asks to range oneself according to opinion about "Competition is good, it stimulates people to work hard and develop new ideas", vs. "Competition is harmful, it brings the worst in people". Range: 1 = Competition is good, 10 = Competition is harmful.)
- Importance to teach a child independence: % of all respondents of a country who mentioned that "Independence" is an important quality children shout learn at home. (They were given a list of qualities that children can be encouraged to learn at home. They should choose up to five they consider to be especially important.)

However, this component is problematic, as it has a Kaiser-Meyer-Olkin measure of sampling adequacy of 0.517. Therefor we run regression with and without competition/merit.[9]

[9] We also run regression with "Competition is a good thing" only. However the results were quite similar.

3.3.2.4 Attitudes towards Novelty (New Ideas and Scientific Progress)

Fianlly, data on attitudes towards novelty, namely on a preference for new ideas and scientific progress also come from the 1990, 1995/1998 and 1999/2000 waves of the World Values Survey. We apply the following:

- "prefer new ideas" is the % of all respondents of a country preferring new ideas over old ones by ranging from 8 to 10. (The survey asks to rate oneself on a scale about "Ideas stood test of time better" vs. "New ideas better", with 1 = Ideas that stood test of time are generally best, up to 10 = New ideas are generally better than old ones.)
- "science advance help" is the % of all respondents of a country saying that scientific advances we are making will help mankind. (The question is "In the long run, do you think the scientific advances we are making will help or harm mankind?" Possible answers: 1 Will help, 2 Will harm, 3 Some of each.)

3.3.3 Data on IPR Protection and Regulation

In order to evaluate the degree of regulation, we use the Economic Freedom of the World Index (Gwartney et al. 2008). The report offers an index of regulation, called "Area 5: Regulation of Credit, Labor, and Business". This index is built upon several sub-indices measuring credit market regulations, labor market regulations, and business regulations (Gwartney et al. 2008, p 189ff). We use it in order to measure the degree of regulation in a society in 2006, and denote this variable by "degree of regulation".[10]

With respect to IPR, we use of one of the sub-indices of Gwartney et al. (2008) Gwartney et al. (2006) belonging to the property right section: the sub-index of the protection IPR ("2C Protection of intellectual property") for the year 2004, the latest IPR-data available. This IPR sub-index is based on data from the Global Competitiveness Report of the World Economic Fo-

[10] The original measure is indexing de-regulation (the lower the degree of regulation the higher the score). In order to avoid confusion we use an inverse version such that highly regulated countries have high scores.

rum.[11] We denote this index by “IPR protection”. Another measure related to IPR are the figures about the software piracy rates in 2006, taken from the Fifth Global Software Piracy Study (Business Software Alliance, 2007).

3.4 Empirical Results

To test hypotheses H1 through H8, we run linear regression models (OLS), varying the endogenous variable as well as the set of explanatory variables.[12] The results appear quite robust and are displayed in Tables 3.1 through 3.3. All three tables are structured as follows: After the control variables, the influence of the variables presenting hypotheses H1 through H8 is shown. We present the three most representative models (for each table equations I, II and III respectively), each with and without "competition/merit" (indicated by a and b). While other variables are skipped in single models, social trust, internet users and IPR are used across the board.[13] We are able run regressions with up to 70 countries, and we are able to distinguish with respect to the level of activities.

Table 3.1 presents the regression results for the number of OSS developers per 1,000 inhabitants including those localized using the information about the SLD. In the Appendix A the reader can find the same regressions for OSS developer data without those localized using the information about the SLD. In a second step, we run regressions for the active OSS developers per 100 inhabitants, again including those localized via SLD, see Table 3.2. For the results without SLD see Appendix A. Finally, we analyze the OSS activity level (Table 3.3). As usual we present the results with those located via SLD while the other version can be found in the Appendix.

[11] Question: IPR protection “in your country is 1 = weak and not enforced, up to 7 = strong and enforced”.

[12] In order to deal with possible heteroskedasticity we run the regressions with robust standard errors, i.e. heteroskedastic-consistent estimates. Furthermore, we check for possible problems with multicolinearity by looking at the pairwise correlations, and also checking the Variance Inflation Factors.

[13] We present results with IPR protection only. We also run regressions with software piracy rates. It turns out that the piracy rates are never significant, while the rest remains basically the same, only regulation becomes insignificant.

Table 3.1: OSS Developers per 1,000 inhabitants (with SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-0.420	-0.389	0.982	0.827	-0.445	-0.430
	(0.773)	(0.788)	(0.341)	(0.455)	(0.768)	(0.773)
education	0.198	0.200	0.312	0.230	0.0557	0.0567
	(0.636)	(0.632)	(0.386)	(0.457)	(0.890)	(0.886)
inet users	1.372***	1.370***	1.016*	1.007*	1.539***	1.537***
	(0.008)	(0.008)	(0.060)	(0.054)	(0.003)	(0.003)
prefer new ideas	1.683	1.518			0.657	0.584
	(0.555)	(0.569)			(0.827)	(0.838)
science advance help	1.254**	1.220**			1.169**	1.154**
	(0.018)	(0.025)			(0.031)	(0.039)
self-determ/indiv	1.168*	1.147**	0.573	0.685	1.229**	1.219**
	(0.051)	(0.047)	(0.290)	(0.238)	(0.046)	(0.040)
intpersonal trust	1.168**	1.174**	0.650**	0.592*	1.162**	1.165**
	(0.013)	(0.012)	(0.046)	(0.071)	(0.024)	(0.023)
competition/merit	-0.0676		0.337		-0.0310	
	(0.831)		(0.269)		(0.916)	
IPR protection	0.373	0.375	0.368	0.356	0.512	0.513
	(0.272)	(0.256)	(0.267)	(0.287)	(0.129)	(0.122)
degree of regulation	-1.048*	-1.042*	-0.852*	-0.828		
	(0.091)	(0.082)	(0.085)	(0.108)		
_cons	-0.988**	-0.975**	-0.318	-0.214	-1.159**	-1.152**
	(0.038)	(0.049)	(0.328)	(0.456)	(0.017)	(0.022)
N	60	60	70	70	60	60
adj. R^2	0.815	0.818	0.785	0.784	0.805	0.809
d. of freedom	49	50	61	62	50	51

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Note that GDP is per 100,000 inhabitants and inet users is per 10,000 inhabitants. Education refers to the year 2002, while IPR protection refers to 2004. The measures self-determ/indiv and competition/merit are based on principal component analysis.

Table 3.2: Active OSS Developers per 100 Inhabitants (with SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-1.358	-1.370	2.820	2.431	-1.415	-1.468
	(0.660)	(0.655)	(0.258)	(0.367)	(0.660)	(0.647)
education	0.499	0.499	0.535	0.330	0.166	0.162
	(0.582)	(0.577)	(0.496)	(0.622)	(0.849)	(0.848)
inet users	2.434**	2.435**	1.808	1.786	2.823**	2.829**
	(0.029)	(0.027)	(0.119)	(0.110)	(0.011)	(0.010)
prefer new ideas	4.202	4.267			1.805	2.070
	(0.480)	(0.448)			(0.774)	(0.732)
science advance help	2.148**	2.161**			1.951*	2.006*
	(0.037)	(0.039)			(0.075)	(0.072)
self-determ/indiv	2.702**	2.710**	1.283	1.565	2.845**	2.881**
	(0.036)	(0.031)	(0.285)	(0.226)	(0.031)	(0.026)
intpersonal trust	2.594***	2.592***	1.214*	1.069	2.580**	2.570**
	(0.007)	(0.006)	(0.060)	(0.110)	(0.017)	(0.016)
competition/merit	0.0265		0.842		0.112	
	(0.970)		(0.205)		(0.862)	
IPR protection	1.466*	1.465*	1.154	1.124	1.791**	1.788**
	(0.057)	(0.053)	(0.131)	(0.148)	(0.021)	(0.020)
degree of regulation	-2.450**	-2.452**	-2.024**	-1.962*		
	(0.044)	(0.039)	(0.033)	(0.052)		
_cons	-2.024**	-2.030*	-0.631	-0.369	-2.425**	-2.448**
	(0.048)	(0.056)	(0.357)	(0.530)	(0.022)	(0.025)
N	60	60	70	70	60	60
adj. R^2	0.829	0.832	0.802	0.799	0.817	0.820
d. of freedom	49	50	61	62	50	51

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Note that GDP is per 100,000 inhabitants and inet users is per 10,000 inhabitants. Education refers to the year 2002, while IPR protection refers to 2004. The measures self-determ/indiv and competition/merit are based on principal component analysis.

Table 3.3: OSS Activity Level: Messages per 10,000 Inhabitants (with SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-1.696	-1.655	2.543	2.286	-1.731	-1.716
	(0.365)	(0.374)	(0.245)	(0.321)	(0.359)	(0.364)
education	0.571	0.573	0.471	0.336	0.363	0.364
	(0.472)	(0.468)	(0.491)	(0.582)	(0.641)	(0.636)
inet users	1.590*	1.587*	1.179	1.164	1.834**	1.832**
	(0.052)	(0.052)	(0.150)	(0.139)	(0.026)	(0.025)
prefer new ideas	3.759	3.548			2.259	2.182
	(0.411)	(0.407)			(0.633)	(0.630)
science advance help	1.253*	1.210*			1.130	1.114
	(0.066)	(0.071)			(0.111)	(0.107)
self-determ/indiv	2.767***	2.740***	1.562*	1.748*	2.856***	2.846***
	(0.006)	(0.005)	(0.088)	(0.075)	(0.006)	(0.004)
intpersonal trust	2.662***	2.670***	1.346***	1.251**	2.653***	2.657***
	(0.000)	(0.000)	(0.009)	(0.020)	(0.001)	(0.001)
competition/merit	-0.0863		0.556		-0.0329	
	(0.895)		(0.292)		(0.957)	
IPR protection	1.019*	1.022*	0.531	0.510	1.222**	1.223**
	(0.091)	(0.081)	(0.397)	(0.429)	(0.039)	(0.036)
degree of regulation	-1.533*	-1.525*	-1.234*	-1.194		
	(0.094)	(0.078)	(0.067)	(0.103)		
_cons	-1.587*	-1.570*	-0.538	-0.366	-1.838**	-1.831**
	(0.070)	(0.081)	(0.356)	(0.479)	(0.042)	(0.046)
N	60	60	70	70	60	60
adj. R^2	0.842	0.845	0.809	0.808	0.836	0.839
d. of freedom	49	50	61	62	50	51

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Note that GDP is per 100,000 inhabitants and inet users is per 10,000 inhabitants. Education refers to the year 2002, while IPR protection refers to 2004. The measures self-determ/indiv and competition/merit are based on principal component analysis.

3.5 Comparison and Interpretation of the Results

In this section we compare and interpret the results of the different estimation models. The control variables do not contribute to the explanation of OSS activities. The evidence for the other variables is mixed, as the following discussion of the variables in the order as introduced in the hypotheses shows.

The share of internet users is positively correlated with OSS activities and significant, which supports hypothesis 1. As already mentioned, this can be interpreted with respect to two aspects. First, internet access is a precondition for participating in OSS development. Second, the number of internet users is a proxy for the size of the ICT sector, which has a positive impact on the supply side of OSS via OSS reputation signals for ICT-job-markets and the potential market size for OSS business models.

Surprisingly "prefer new ideas" is positively but not significantly correlated with OSS activities. Thus, hypothesis 2 has to be rejected, as country-wide openness to new ideas is not encouraging participation in OSS. Interestingly, this is different with respect to the attitude towards scientific progress (hypothesis 3): A positive attitude towards scientific progress (science advances help) is clearly significant with respect to the number of developers. It is also significant with respect to active developers, and the activity level where it has its lowest significance level. The preference for new ideas and the attitude towards scientific progress measure different aspects (someone who likes new ideas can still be skeptical about the impact of scientific progress). Nevertheless one might expect that it is the combination of both that is beneficial for OSS. The argument would be that openness to new ideas has to meet a preference for scientific aspects in order to fit into the OSS community. Therefore, we also run regressions with an interaction term, but this is never significant.

As for hypothesis 4, stating that the degree self-determination of a society has a positive impact on the number of OSS developers as well as on the OSS activity level, the number of (active) OSS developers at SourceForge is indeed positively correlated with the degree of individualism. Interestingly the significance level rises when it comes to the activity level (see Table 3.3). This fits our expectations. OSS development can be a way of individualistic

self-fulfillment. Therefore, it is highly plausible that societies with high account of self-determination are experiencing a higher OSS activity level.

The number of OSS developers is positively correlated with the degree of interpersonal trust. Therefore hypothesis 5 is not rejected, fitting also our expectations. Again, despite the fact that this variable is highly significant throughout all equations, it is interesting to notice that this factor is more significant when applied to the activity level. In a society generating mutual trust, private provision of public goods indeed seems more likely.

A culture of positive attitudes toward competition and the merit principle is not relevant: the principal component "competition/merit" was never significant (nor was the positive attitude towards competition solely). Therefore hypothesis 6 has to be rejected. A possible explanation could be that individualistic self-fulfillment aspects and self-determination are more important on the level of culture.

How do IPRs affect OSS activities? In the regressions of model I and III the number of active OSS developers and the level of activity both are positively correlated with the degree of protection of IPR. Hence, hypothesis 7 cannot be rejected. It seems that indeed the supply side of OSS benefits from the security of IPRs. This is plausible if one remembers that, as already mentioned, OSS licenses are build upon copyright law, OSS projects use trademark law, etc. OSS relies on the idea of IPR, although it uses this institution in a new way. The deny of IPR as such might even harm the supply of OSS. We have to discuss an often mentioned objection in this context here: The argument would be that in societies with a low de facto protection of IPRs there is not so much need for OSS, as one can get software for free (or at least at low costs) anyway. This shall explain why we have more OSS contribution when IPR protection is strong. Hence, this argument sees OSS as a substitute for pirated software. However, this explanation is not convincing because of various reasons. First, if OSS is a substitute for piracy software we should see an effect of piracy rates. But piracy rates are never significant (see footnote 13). Second, OSS is far more than just "cheap" software: the key element of OSS is that one has access to the source code and can thus further develop it etc. A pirated copy of proprietary software (closed source software) is still just a copy of the binary code. Whereas the source code is the human-readable recipe of software, the binary code is not readable by humans. Thus piracy software cannot be a substitute for OSS

as it is missing the source code. Finally, the objection mixes up demand- and supply side arguments. The argument points to the demand-side, but we analyze the supply side of OSS: the number of (active) developers and the activity level.

In most regressions with regulation the variable has a negative sign and is significant. Thus we find support for hypothesis 8. OSS activities obviously depend on regulations, exactly as other entrepreneurial activities and individual initiatives are positively correlated with lower regulation, i.e. a set of reasonable regulations.

Positive attitudes toward competition and the merit principle, protection of IPRs, and less regulation are all aspects that relate to OSS and OSS business models. One might argue that it is precisely the combination of less regulation and a sense for competition, or less regulation and good IPR protection, or even the combination of the three, that is in favor of business and thus also of OSS business models. This could then affect the OSS activities as well. Therefore we also run regressions with such interaction terms. The results are clear: only the interaction term of IPR protection and regulation is significant (positively) in some regressions. It has an impact only if we examined the activity level and the active developers.

3.6 Summary

This chapter presents a cross-country study of how the relative number of OSS developers and the OSS activities of a country depend on institutional and cultural factors that belong to level one and two of Williamson's framework. For this purpose we break up the phenomenon OSS into several elements, identifying more general, underlying aspects. We then connect these aspects with institutional and cultural factors. In detail we have examples of the impact of culture in terms of values and preferences and in forms of beliefs, and of how culture can foster or hinder the implementation and/or functioning of institutions. In addition we have examples of how formal 'level two'-institutions have an impact on the functioning of lower-level formal and informal institutions (IPR protection) and on the outcome, thus payoffs, of certain activities.

We can assign 1.3 million OSS developers from SourceForge to their countries, that equals 94% of the total at SourceForge in 2006. With the posted messages we have a proxy for the activity of each developer. We are able to run regressions with about 70 countries, as the data about the cultural and institutional variables are not available for all the countries existing.

Beside the fact that internet access is an important factor for the supply side of OSS, our findings indicate that a positive attitude towards scientific progress as well as a culture of self-determination/individualism is in favor of OSS. The same is true for interpersonal trust. IPR protection is significant in some regressions, always with a positive sign. And finally less market regulation fosters OSS.

Our analysis shows that the non-equal geographical distribution of OSS activities is not driven by aspects like GDP and education, and only partly by internet access. It can mainly be explained by the differences in several cultural and institutional factors. This underlines the importance of these factors. Hence this study shows the impact of such factors on micro(economic) behavior, using the case of OSS development as a special example. This can help to better understand the role such cultural and institutional factors play. But it also improves the understanding of the phenomenon OSS. Our findings support a view of OSS as being an entrepreneurial activity that relies on trust as well as on IPR protection. It has a strong individualistic/self-deterministic aspect, combined with a spirit of individual initiatives. The fact that OSS can be the basis of a business model is also supported by our findings, as the results for market regulation and IPR protection show. Finally, the fact that the supply side of OSS benefits from the security of IPRs indicates that OSS is not anti-IPR but a new IPR paradigm. This IPR regime coexists with CSS. In the next chapter we therefore concentrate on the rationale for the open and the closed source principle using a property rights approach.

4 Intellectual Property Rights and Ex-Post Transaction Costs

> "If the main allocative function of property rights is the internalization of beneficial and harmful effects, then the emergence of property rights can be understood best by their association with the emergence of new or different beneficial and harmful effects."
>
> *H. Demsetz, 1967, p 350*

4.1 Introduction*

In Chapter 3 we have seen that OSS activities are positively correlated with the degree of protection of IPRs. OSS and CSS are both based on copyright law. However, there is a difference in the scope of rights transfered by the OSS vs. CSS licenses. Thus, OSS and CSS are different ownership concepts with different IPR allocations and different modes of organization. In other words: OSS and CSS represent different strategies for using the resource software, i.e. its source code. Both have specific advantages and drawbacks at the individual and firm level as well as for social welfare.

This chapter focus on these two different IPR regimes. We use a property rights perspective to examine and explain the rationale for OS and CS principles in a general way. Therefore, this chapter contributes to the institutional literature on OSS,[1] to the property rights theory in general and to the literature on how the OS paradigm can be transferred to other industries. The analysis of this chapter is a unique approach. The only contribution that links the phenomenon of OSS to the concept of transaction costs is Benkler (2002). Benkler refers to Demsetz's explanation of the emergence of property rights and Coase's theory of the firm (Demsetz,

*This Chapter is based on von Engelhardt (2008b)

[1]For example Weber (2004b); Brand and Schmid (2005); Gehring (2006), see Section 1.3.2 for details.

1967; Coase, 1937). However, Benkler (2002)'s approach differs from the analysis of the current chapter. First, he refers more generally to "commons based peer production". Second, Benkler sees the peer cooperations based on commons without private property (or: in the "absence of property"). In contrast to this, we will argue that OSS is based on private property, although the amount of exclusive ownership is reduced. Third, Benkler focuses on the information problem of who is the best person for a given talk. His main argument is that under certain circumstances peer production is better suited than markets or hierarchies (firms) to assign the right "talent" to a task. This might explain why common based peer production is successful. It does not necessarily explain why individuals choose this. The theoretical framework developed in this chapter focuses on why rational individuals choose both, OSS and CSS. Based on a property rights approach the rationale for both IPR paradigms is discussed. The argument is that OS and CS regimes coexist because both are second-best solutions.

Property rights theory tends to concentrate on negative external effects, i.e. *scarce* resources. Probably the best known examples are the "tragedy of the commons" (Hardin, 1968) and the "tragedy of the anti-commons" (Heller, 1998). This focus can be traced back to Demsetz' seminal article. Although he mentions the "internalization of external costs and *benefits*" (Demsetz, 1967, p 349, emphasis added) he focuses entirely on negative externalities. By contrast, this chapter considers a *non-scarce* and even *anti-scarce* resource. Furthermore, some property rights regarding the non- and anti-rival applications of the resource 'source code' are de facto not separable because of ex-post transaction costs and the economic characteristics of software. In this case, it may be rational to adopt both OS and CS.

Section 4.2.1 introduces our analytical framework. Section 4.2.2 explains how the economic characteristics of software make source code a non-scarce and even anti-scarce resource. We will see, that neither non-rivalry nor anti-rivalry alone can explain the dichotomy between OSS and CSS. This is because a perfect market would lead to an optimal allocation of non- and anti-rival applications. Section 4.2.3 defines the optimal allocation of property rights and the corresponding licensee agreements. Following Coase we then ask what kind of transaction cost problems limit market transactions such that this leads to the coexistence of CSS and OSS. We show in 4.3.1 that the OSS-CSS phenomenon cannot be explained

by limited internalizability of some positive effects but because of *ex post transaction costs* however, some property rights are de facto *non separable*. This leads to a control problem, i.e. a de facto dilution of exclusive ownership. CS and OS present two different solutions of this problem. Whereas CSS maximizes control and exclusive ownership, OSS minimizes it (Section 4.3.2).

4.2 Intellectual Property Rights and a Non- and Anti-Scarce Resource

4.2.1 The Analytical Framework

This section introduces the notation and definitions used in this chapter.[2] We use set theory to define an economic resource and its subsets. Based on this we define the terms 'scarce', 'non-scarce' and 'anti-scarce'.

4.2.1.1 An Economic Resource

Resources can be generally defined in terms of a technically meaningful set of certain elements. Source code in particular can be defined in terms of code lines. We therefor describe a given source code as a set X. X can be split up into subsets of code lines and there exists a set of all subsets $\mathcal{P}(X) = \{A \mid A \subseteq X\}$.

Let $y = f(Z, \cdot)$ denote the $Z \in \mathcal{P}(X)$ used in application y. The 'use' $f(Z, \cdot)$ is one of several possible transformations of Z, either by itself ($y = f(Z)$) or in conjunction with *other* code lines. For example, $y = f(Z, W)$ would be an application of the combined code line sets Z and W. This can be rewritten as $y = f(V)$ with $V = \{Z \cup W\} \neq \emptyset$.

For technical reasons, applications do not exist for all Z. Here the trivial example is $Z = \emptyset \in \mathcal{P}(X)$. Hence, there exists a set of technically *non*-meaningful subsets of X: $\mathcal{U}(X) = \{Z \in \mathcal{P}(X) \mid \nexists\, y = f(Z, \cdot)\}$. And finally:

[2] Although this analytical framework was developed to analyze 'source code', it can be applied to *any* economic resource.

Definition 4.2.1. The set of technically meaningful subsets of X is

$$\mathcal{X}(X) = \{\mathcal{P}(X) \setminus \mathcal{U}(X)\} = \{Z \in \mathcal{P}(X) \mid \exists\, y = f(Z,\cdot)\}. \tag{4.1}$$

Note that Z can have multiple applications, i.e. there can exist several $Z \in \mathcal{X}$ with $\exists!\, \mathbf{y} = (y^1, \ldots, y^n)$, $y^i = f(Z,\cdot)$, $n \geq 2$.

Definition 4.2.1 makes it possible to define a corresponding set of applications for each X which we denote by Y:

Definition 4.2.2. The corresponding set of applications of X is

$$Y(X) = \{y \mid y = f(Z,\cdot), Z \in \mathcal{X}(X)\}. \tag{4.2}$$

As previously mentioned, $y = f(Z,\cdot)$ indicates that Z might be but does not necessarily have to be combined with other code lines. Therefore, a more general notation is $y = f(Z,\cdot) \in \{f(Z), f(V)\}$ with $V = \{Z \cup W\} \neq \emptyset$ and $Z \in \mathcal{X}(X)$. This yields the following general notation for the corresponding set of applications: $Y(X) = \{f(Z), f(V)\} = Y(Z) \cup Y(V)$.

4.2.1.2 Definition of Scarce, Non- and Anti-Scarce

A *scarce* resource is a resource with *rivalry in use*. Formally, scarcity occurs if the use of $Z \in \mathcal{X}$ for any application $\check{y} \in \check{Y}$ leads to *rivalry in use*.

Definition 4.2.3. X is called a *scarce resource* with respect to $\check{Y} \subseteq Y$ if

$$\forall\, \check{y} = f(Z,\cdot) \mid \left(Y^{new}(X^{new}) \subset Y(X)\right). \tag{4.3}$$

A *non-scarce* resource is a resource that is *non-rival in use*, i.e. a public or club/toll good. In this case, the use of $Z \in \mathcal{X}$ for any application $\tilde{y} \in \tilde{Y}$ does *not* lead to *rivalry in use*.

Definition 4.2.4. X is called a *non-scarce resource* with respect to $\tilde{Y} \subseteq Y$ if

$$\forall\, \tilde{y} = f(Z,\cdot) \mid \left(Y^{new}(X^{new}) = Y(X)\right). \tag{4.4}$$

An *anti-scarce* resource is a resource with *anti-rivalry in use*, i.e. the more the resource is used the higher is its value. This happens because use increases the set of applications.[3] Thus, a resource is anti-scarce, if the use of $Z \in \mathcal{X}$ for any application $\hat{y} \in \hat{Y}$ leads to *anti-rivalry in use*.

Definition 4.2.5. X is called an *anti-scarce resource* with respect to $\hat{Y} \subseteq Y$, if

$$\forall\, \hat{y} = f(Z, \cdot) \mid \left(Y^{new}(X^{new}) \supset Y(X) \right). \tag{4.5}$$

4.2.2 The Resource Software

4.2.2.1 The Economic Characteristics of Software

Software has specific economic characteristics. This can be summarized as follows (for more details see von Engelhardt, 2008a):

- Software is a *digital good* and therefore (re)combinable: software products are "cumulative and emergent—new digital goods that arise from merging antecedents have features absent from the original, parent digital goods" (Quah, 2003, p 19). We will define the terms *recombinable* and *combinable* as follows: Let $\mathcal{S}(X)$ denote the set of all permutations of X, and $S \in \mathcal{S}(X)$ denote one permutation of X. X is called *recombinable* if $\exists\, S \neq X$ s.t. $\exists\, y = f(S)$. Of course, it is unrealistic to think that one can build new software simply by rearranging code lines. In most cases, all or parts of the given source code will have to be *combined* with other code lines.

  Definition 4.2.6. $Z \in \mathcal{X}$ is called *combinable* with respect to W, if

  $$\exists\, W \neq \emptyset \text{ s.t. } \exists\, y = f(V),\ V = \{Z \cup W\} \neq \emptyset. \tag{4.6}$$

- Software is *aspatial*, i.e. infinitely expansible and therefore nonrival. Once software is produced, it can be reproduced without any loss of quality at negligible costs. On the other hand, it has high development and pre-launch testing costs (first copy costs). These combination of high sunk costs and low marginal costs leads to a subadditive cost function.

[3] Note that the term "applications" captures both qualitative and quantitative changes in the resource.

- Software is a *network good* with direct and indirect network effects.[4] Network effects are intimately connected with *complementarity* which "means that consumers in these markets are shopping for *systems* [...] rather than individual products" (Shy, 2001, p 2, emphasis original). Furthermore, *modularity* plays an important role (Weber, 2004b, pp 172 ff; Langlois, 2002, pp 22 f). Compatibility is a necessary condition for 'complementarity' and 'modularity'.

- Compatibility and combinability are necessary (but not sufficient) conditions for yet another software characteristic: *cumulativeness*. If X is combinable and V is compatible to X, then X can be cumulative. X is cumulative if V itself is part of the new X^{new}.

  Definition 4.2.7. X is called *cumulative* with respect to W, *if*

  $$\exists W \neq \emptyset \text{ s.t. } \Big(\exists y = f(V), V = \{Z \cup W\} \neq \emptyset\Big) \wedge \Big(Z, V \subseteq X^{new}\Big) \quad (4.7)$$

- Software is an *information good* because its source code is information, i.e. a human-readable recipe. But software differs from other information goods: some consumers do not care about the information but merely about its capabilities.[5] This explains why the software is an information good that can be sold in a state users cannot read the information. CSS is typically distributed in machine-readable binary codes that leave the information 'closed'.

4.2.2.2 Software as a Non- and Anti-Scarce Ressource

Because of softare's economic characteristics, a source code is a *non-scarce resource*. Obviously there is no rivalry in consumption. However there is also no rivalry in production:

[4] Software is designed to process data. Hence, it must exchange data with other software (applications, operating system) and/or hardware. This requires compatibility and leads to network effects. Two different types of network effects play a role: (1) the installed base effect, i.e. utility increases with the total number of users; (2) the personal network effect (von Westarp, 2003), in which users adopt based on the adoptions in the personal network.

[5] For example, the vast majority of people who use software to receive and send emails have no desire to read the source code.

(i) Because of near-zero reproduction costs, the first copy X is a non-scarce resource for producing $n+1$ copies. Since X can be copied without any loss of quality is $y :=$ 'copying' $\mid Y^{new} = Y$.

(ii) Because it is combinable and does not deteriorate, any given source code X is a non-scarce resource for further software development. The code X, or parts of it, can be used as input to develop first-copies of new software products:[6] $y = f(V) \mid Y^{new} = Y$.

(iii) Since software is an information good, any given source code X is a non-scarce resource for knowledge spillovers, i.e. transfering ideas and learning. Source code is a list of programming solutions. Existing code may contain several solutions that would be useful in a new software project. Such a 'transfer of ideas and concepts' is even possible from one programming language to another. Additionally, a software engineer can improve his or her programming skills from reading a source code (learning). Hence, for all applications y that transfer ideas or imply learning is $y = f(Z) \mid Y^{new} = Y$.

In addition of being a non-scarce resource software can sometimes display anti-rivalry[7] in use, making it an *anti-scarce resource*

(iv) Because of standards and network effects, software is a network good (e.g. see White et al., 2005; Kooths et al., 2003; Gröhn, 1999; Gandal, 1994). If X is a network good for a set of 'network applications' $y^{nw} \in Y$, then the following holds: $\forall y^{nw} \mid Y^{new} \supset Y$. This means that X is an anti-scarce resource with respect to y^{nw}, see the definition 4.2.5, p 75.

This is true for all network goods. One intuitive example is a network of telephone lines. If more users plug into to the network, more applications (telephone connections) are possible.

[6] Because of this there are whole catalogs of complete elements of programs (Gröhn, 1999, p 5) and a distinct programming approach—the so-called component-based software engineering—emerged. This approach also includes the re-use of software components across producers (Romberg, 2003, pp 253 ff.).

[7] Compare the following also with Weber (2004b, pp 153 ff.), where one can find similar thoughts. Weber uses the term 'antirivalness'.

(v) Because of cumulativeness, the use of a source code dislays anti-rivalry: If X is cumulative with respect to any $W \in \mathcal{W}$, then X is an anti-scarce resource with respect to $\mathcal{W}$.

Proof: From cumulativeness (definition 4.2.7, p 76) we obtain

$$\begin{aligned}
&\forall W \in \mathcal{W} \mid \left(\exists\, y^c = f(V), V = \{Z \cup W\} \neq \emptyset\right) \wedge \left(Z, V \subseteq X^{new}\right) \\
&\Rightarrow \left((V \nsubseteq X) \wedge (V \subseteq X^{new})\right) \wedge \left((y^c \notin Y) \wedge (y^c \in Y^{new})\right) \\
&\Rightarrow \forall y^c = f(V) \mid \left(Y^{new}(X^{new}) \supset Y(X)\right),
\end{aligned}$$

which is identical to the definition of anti-scarcity (definition 4.2.5, p 75).

If a software engineer further develops a module—e.g. because of own needs—others can benefit from this improvement provided that the new piece of source code is still compatible and the new 'module' is implemented in users' software systems. Thus, a source code is a potential input for further software development. The new output again is (potential) input for further software-development, and so on.

4.2.3 Optimal Allocation and Optimal Licenses

We will now discuss IPRs for a non- and anti-scarce resources in a world without transaction costs. Since this chapter focuses on software (source code), IPR allocation is defined by licenses. Software is traditionally protected by copyright law (Graham and Somaya, 2004, p 269). The copyright-based license agreements define the transfer of the rights. We will see that there exists an optimal allocation of rights regarding non- and anti-rival applications, i.e. optimal licenses can be defined.

Defining IPRs for *non-rival applications* refers to the private provision of a good without rivalry in use, i.e. a club good. Standard microeconomics teaches that a commodity without rivalry in use should be supplied if the sum over each individual's willingness to pay (WTP)equals or exceeds the total costs (C), i.e. if $\sum_{j=1}^{m} WTP_j \geq C$, with m agents.

Let us assume, that there are a finite number of applications $y^i \in Y$, $i = [1 \ldots n]$, each of which is traded in a different market. We also assume that

(i) the owner of the resource can discriminate in price, such that each agent pays an individual price for the application i denoted by p_j^i

and

(ii) each of the n markets is a contestable market, such that the incumbent must choose the lowest price-vector that covers his or her costs.[8]

This yields

$$\sum_{i=1}^{n}\sum_{j=1}^{m} p_j^i = C, \;\; p_j^i \leq WTP_j^i. \tag{4.8}$$

This private provision of the resource clearly maximizes welfare.

Optimal provision is also possible for the the rights to use the *anti-rival* applications. It is well-known in network theory that ownership can internalize network effects. Provided that it is possible to price every single plug-in, network effects can perfectly be internalized by dynamic pricing, i.e. by individual price discrimination that takes into acount the marginal benefits of adoption (Liebowitz and Margolis, 2002, 1994; Katz and Shapiro, 1994). Since optimal internalization requires perfect price discrimination with respect to adoption, each adoption must be traded separately. This can be applied to *any kind* of *positive feedback* mechanism, i.e. to all *anti-rival applications*. We therefore conclude that welfare maximizing allocations are achievable whenever the source code owner can perfectly price discriminate.

Consider n markets where y^is (the applications) are traded and m agents pay $p_j^i \geq 0$ for each $y^i \in Y$. A price $p_j^i = 0$ implies that agent j has a zero *WTP* for application i and does not buy it.[9] Thus, a license agreement is optimal if it transfers a set of IPRs to agent j that includes all applications for which the agent would pay a positive price. Recall that X denotes the resource, and Y denotes the corresponding set of applications, with $y^i \in Y$.

[8] Assuming perfect contestable markets in context of software my be problematic. However, the argument does not change in the presence of market power. While the monopolist or oligopolist gains an extra profit, perfect price discrimination ensures that welfare is maximized in this case also.

[9] In principle, one could also allow for 'negative prices', thus $p_j^i < 0$. This would imply that the agent gets paid for activities that yield cumulative benefits, for example. However, we restrict the analysis to $p \geq 0$ for ease of exposition.

Furthermore, let us denote[10] an IPR by h, and a set of IPRs by $\mathbf{h}_j$ Then the condition for an optimal license is formally given by

$$\mathbf{h}_j : X \to \{y^i \mid p_j^i > 0\}. \tag{4.9}$$

The price of the license is

$$p_j = \sum_{i=1}^{n} p_j^i = (p_j^1, \ldots, p_j^n) \cdot \mathbf{1}^T, \tag{4.10}$$

where $\mathbf{1}^T$ is the transpose of the one-vector.

Based on the foregoing, we see that it is possible to derive the welfare maximal allocation of non- and anti-rival applications. Furthermore, a perfect market with perfect price-discrimination would lead to this allocation. Based on this, optimal licensee agreements can be defined. This would lead to an optimal allocation of IPRs.

4.3 The Role of Transaction Costs

In a world with complete information and knowledge rational individuals would trade each single $y \in Y$. However, transaction costs lead to incomplete information and/or knowledge. This explains why property rights are defined and bundled in license agreements which are then traded. If agents only know $Y_i' \subset Y$ it makes sense to trade 'rules' (i.e. licenses) rather than single applications. Incomplete knowledge changes the neoclassical standard market game into a Bayesian market game where agents define, trade and price the rights based on their subjective expectations of Y. And there exist Such rights often enable an innovator to appropriate at least some of the created benefits from a new use of a resource. This provides incentives to search for innovative usage, and can lead to competition as a discovery procedure (von Hayek, 1978).

The question that remains is: why does OS *software* establish? Which problem of internalization leads to the dichotomy between OSS and CSS?

[10] We use a simplified right notation based on set theory. See Appendix B for a more detailed definitions and notation.

4.3.1 Incomplete Information, Transaction Costs and Limits of Internalizability

Some transaction costs limit the internalizability of software's positive external effects. However, we will see that such limits cannot explain the coexistence of OSS and CSS. We start by exploring network effects and cumulative effects that produce anti-rivalry.

As already mentioned before, property rights can internalize network effects. In principle this proposition still holds in a world with transaction costs, because a new member increases the value of the network once and only once at the moment of joining. As this 'plug-in' is connected to a right (the right to join the network) and be thus be priced dynamically.

Internalizing positive externalities from the cumulative effects described in item (v) (p 78) is more complicated. Consider a developer who writes a certain piece of source code. Suppose further that several users have access to this code and want to use it as an input for cumulative work. Perfect internalization requires that the developer can charge each user every time he or she uses the source code as input or changes it. Obviously, transaction costs inhibit this internalization, especially for (very) small improvements and minor changes. This is particularly true for software that is developed cumulatively developed in a series of small steps.

The knowledge spillovers described in (iii) (p 77) also cannot be perfectly internalized. The moment of 'adoption' that causes knowledge spillovers is hard to observe, and also neither the date nor the frequency of future value creation is known. The value of the 'adoption' is also hard to evaluate, particularly when the benefits consist of increased skills that may improve future performance.

Thus, while network effects can be internalized, cumulative effects and knowledge spillovers cannot because the positive external effects of knowledge spillovers and cumulative effects are hard to observe and measure. This lack of internalizability leads to too little of the relevant activity as the social benefit is greater than the private benefit. However, the degree of this 'market failure' is limited. Since individuals can usually form subjective expectations of future benefits they gain from, for example, consuming

reference books.[11] So this phenomenon cannot solely explain the coexistence OSS and CSS because it is not unique to software markets. Lack of internalizability exists in many other markets where we do not observe OS and CS licenses existing side by side.

4.3.2 Ex-Post Transaction Costs and the Problem of Not Separable Rights

4.3.2.1 The Problem

Obviously no one would claim non-enforceable rights. Suppose than some rights are enforceable only if they are bundled. This can happen when rights are formally separable but cannot be enforced in practice if they are unbundled. The reason are ex post transaction costs.

Consider a software-related example in which IPRs *are* separable. Software can be distributed in machine-readable copy-protected versions that let users use the code without copying or changing.[12] More generally, Software-IPRs *are* separable if the applications covered by the right do not require access to the source code. This is true for the applications described in item (i) (n+1 copies) and (iv) (network effects).

Before discussing applications that require access to the source code, we first define non-separable rights. Denote a right which was transferred to agent j by h_j. Let $D_j = D_j(X)$ be the set of applications the right holder j is *de facto* able to do. The right h_j is not separable if the following holds. Applying the right h_j to the resource X yields the set of applications covered by this right Y_j^h (what j is legally allowed to do). Now, there exist at least one application y which j is de facto able to do ($y \in D_j$) but which is not covered by the right h_j: $y \notin Y_j^h$.

Definition 4.3.1. A right is *not separable*, if

$$h_j : X \longrightarrow Y_j^h \subset D_j \;\Rightarrow\; \exists y \in D_j \mid y \notin Y_j^h. \tag{4.11}$$

[11] One can argue that a price of such books includes the expected future gains from knowledge spillovers.

[12] The existence of so-called pirated software shows that legal and technical copy protection is imperfect. Nevertheless, the existence of CS business models shows that enforcement is reasonably effective.

The applications described in item (ii) (source code as input, p 77), (iii) (knowledge spillover, p 77), and (v) (cumulativeness, p 78) require access to the source code. This implies that the corresponding IPRs are de facto *not separable*. If one trades single non-separable rights one has to grant access to the source code which leads to a *club of source code users*. This can affect other rights as the source code may be is used in ways not authorized by the transferred single rights. Club owners normally offer the right to use (usus) the club-good and retain the right to recover the usus from members who do not comply with the rules. Here, however, is a de facto 'dilution of the property rights' (Picot et al., 2005, p 47). Formally the original source code owner can still retain all rights except the usus. As a practical matter, his rights are diluted because of transaction costs. As the number of club members increase, control costs rise so hat it is no longer possible to stop members from using the source code for unauthorized purposes or reselling it. More formally, transaction costs are a function of the number of members (m): $TC = f(m)$ with $\partial TC/\partial m > 0$. Assuming that the resource owner first trades with the agent offering the highest price, the m agents are arranged their willingness to pay, returns ($R(m)$) are a concave function of club members.[13] Figure 4.1 depicts this situation. The optimal number of members corresponds to the dotted line. If the marginal transaction costs are high the optimal m can be small and even zero.

Control costs ceteris paribus *decrease* as the contract becomes less specific, see Figure 4.2. Transaction costs (TC) decrease with less specificity and approach to zero in the limit where every possible application is authorized so that it is unnecessary to control.[14] The question remains whether a market for such unspecific licenses can establish and be stable.

The set of realizeable applications (Y_i^{tc}) of an agent determines her willingness to pay for a completely unspecific contract. The source code owner could therefore perfectly price discriminate if he would *know* each Y_i^{tc}. The problem of course is that the resource owner does have this information. Agents have an incentive to claim that their Y_i^{tc} is smaller than it really is in order to obtain a smaller price. The owner of the source code might know—

[13] The argument also holds, if all agents have the same willingness to pay so that $R(m)$ is linear.

[14] Note that the argument holds regardless of the transaction cost curve is convex or linear.

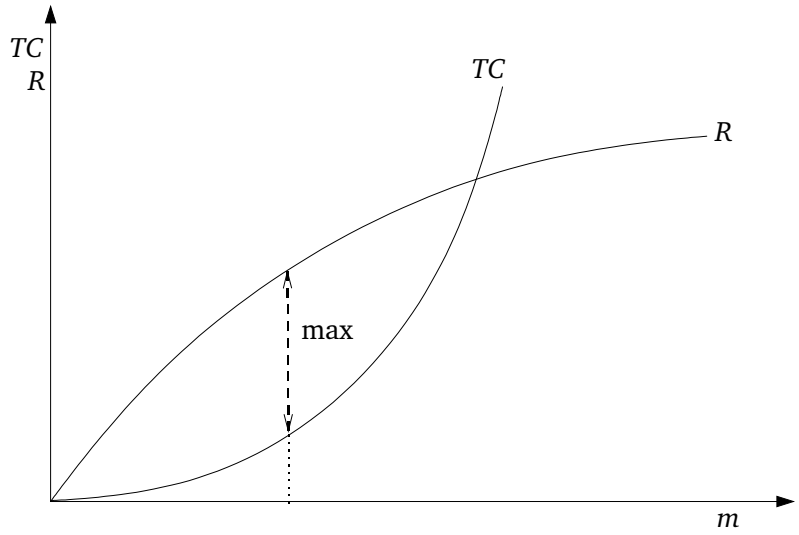


Figure 4.1: Transaction Costs and Limits of Size (Members)

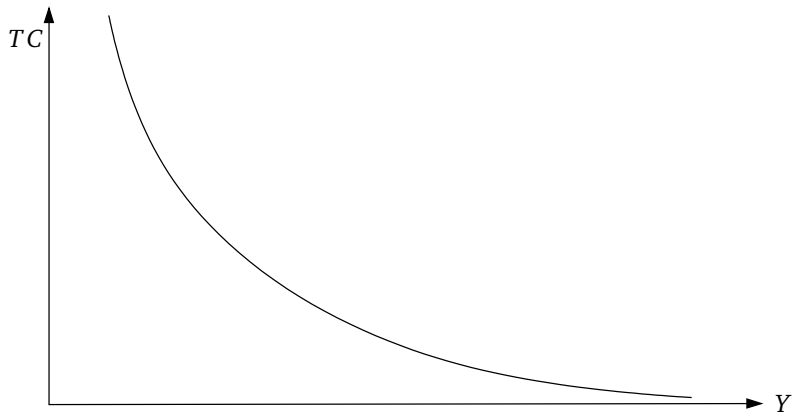


Figure 4.2: Transaction Costs and Specification of Contract

or at least have an approximate idea about—the average set of realizeable and tradeable applications $\bar{Y}_i^{tc}$ respectively. He could thus charge an 'average' price. But given this average price, agents with $Y_i^{tc} < \bar{Y}_i^{tc}$ will decide not to pay this price and drop out the market. This, however, increases the average set of applications and therefore the average price. Again, some agents will drop out the market. This yields an adverse selection process.

We have argued above that because of ex-post transaction costs, IPRs that imply access to source code are de facto not separable. This inhibits the optimal allocation of the IPRs since selling of non-separable IPRs is limited. If one wants to transfer non-separable rights, the whole bundle has to be transferred. Such a non-specific license cannot be sold, but implies low-to-zero transaction costs.

4.3.2.2 Implications for Licensing: The Case of CSS vs. OSS Licenses

A software license transfers a set of rights to the user. As usually (see e.g. Furubotn and Richter, 2005), we distinguish between usus (the right to use a resource), abusus (the right to change a resource) , usus fructus (the right to retain the revenues generated by the resource) and the alientation right (the right to transfer rights of the resource). Existing software licenses can be classified by the scope of transferred rights (see also Hawkins, 2004, p 107; Böhnlein, 2003, p 19 ff and Nüttgens and Tesei, 2000, p 11): Böhnlein, 2003

- CSS licenses are *exclusive* because they are based on the principle of 'closeness'. The code is 'closed', and the user (licensee) typically receives only the usus and a (perhaps restricted) usus fructus from the licensor. The alienation right is not transferred or restricted (see Table 4.1).
- OSS licenses are *inclusive* since they are based on the principle of 'openness'. The licensor *in principle* transfers the whole set of rights to anybody who wants it (see Table 4.1). However, licenses differ in the scope of transferred rights. *Liberal* OSS licenses—like the BSD[15]

[15]BSD stands for 'Berkeley Software Distribution'.

license—do not restrict the use of the software and the source code in any way. *Restricted* OSS licenses—like the GPL[16]—restrict users' alienation rights, since the right to redistribute is limited: Any further developed software as well any derived work must be licensed as a whole under the same type of license, if this new code is further transferred. Hence, it is not true that OSS frees software from property law, indeed e.g. the GPL is based on copyright. An OSS license is a contract that offers everyone all possible rights except for possible restrictions of the alienation right, which must be considered only at the time of redistribution.

Table 4.1: The Transfer of Rights of CSS and OSS Licenses

	Usus	**Usus Fructus**	**Abusus**	**Alienation Right**
CSS	+	+	-	(-)
OSS (restricted license)	+	+	+	(+)
OSS (liberal license)	+	+	+	+

The coexistence of OSS and CSS licenses can be explained by the theory of non-separable rights: CSS licenses are based on exclusive ownership. Only the separable IPRs are traded, and access to the source code is blocked. Conversely, OSS licenses offer the complete set of rights, which is technically mirrored by the access to the source code. Both types of license imply a particular way to organize production and hence governance structure.

The question remains whether it is rational to deliberately forgo some exclusive rights by choosing an OSS license. The next section discusses this issue for non- and anti-scarce resources.

4.3.2.3 On the Rationality Not to Claim all Rights

In some cases the source code owner may be able to forgo some exclusive rights without incurring any real costs. After all, *real* costs of forgoing rights are (only) the lost revenues from the subset of rights that one could

[16]The GPL, the 'GNU General Public License', is the most famous OSS license.

actually have sold. Let $Y_i^{tc} \subset Y$ denote the set of applications an agent i can realize and/or trade Here the superscript tc indicates that the set is bounded by transaction costs. The subscript i indicates that the set also depends on individual 'factors', like the access certain resources. Thus for example, a student's Y_i^{tc} for a certain source code will usually be smaller than Microsoft's Y_i^{tc} for the same code.

We start by defining the role of IPRs for non-scarce resources. Irrational envy and stinginess apart, each agent i should always be willing to forgo those rights where not claiming cause no real costs. This holds if the rights neither belong to Y_i^{tc} nor indirectly affect the value of rights $y^{tc} \in Y_i^{tc}$. Formally, this is defined by $\mathbf{h}: X \longrightarrow \left\{\tilde{y} \notin \tilde{Y}_i^{tc} \mid (\nexists\, y^{tc} \in Y_i^{tc} \mid \varepsilon \leq 0)\right\}$ with ε as the cross-price elasticity of $y^{tc} \in Y_i^{tc}$ regarding $\tilde{y}$.

All other rights should be claimed. This leads to the definition of the *optimal* exclusive IPRs for a non-scarce resource:

Definition 4.3.2. For a non-scarce resource, the rights $\mathbf{h}$ that should be claimed are given by

$$\mathbf{h}: X \longrightarrow \tilde{Y}_i^{\mathbf{h}} = \left\{\{\tilde{y}_i^{tc} \in \tilde{Y}_i^{tc}\} \cup \{\tilde{y} \notin \tilde{Y}_i^{tc} \mid (\exists\, y_i^{tc} \in Y_i^{tc} \mid \varepsilon > 0)\}\right\},$$

with $\pi_i(\tilde{Y}_i^{\mathbf{h}}) = \pi_i(\tilde{Y}_i^{tc})$ and ε as the cross-price elasticity of $y^{tc} \in Y_i^{tc}$ regarding $\tilde{y}$.

This says that exclusive IPRs for a non-scarce resource should be defined so as to protect (a) all applications agent i can *realize and/or trade*, and (b) all applications that are *substitutes* to any y_i^{tc}

We now discuss ant-scarce applications. The argument for the feedback-effects is basically the same. The only difference is that it can now be rational not to claim IPRs in some $\{\hat{y} \notin \hat{Y}_i^{tc} \mid (\exists\, y_i^{tc} \in Y_i^{tc} \mid \varepsilon > 0)\}$ and even some $\hat{y}_i^{tc} \in Y_i^{tc}$, where the benefits from feedback effects exceed the costs. Because of definition 4.2.5 (anti-scarcity, p 75) we know that $\forall\, \hat{y} \mid \left(Y^{new} \supset Y\right), \Rightarrow \exists Y^+ = \{Y^{new} \setminus Y\}, \Rightarrow\ \hat{y} \rightarrow Y^+$. This leads to the following definition of *optimal* exclusive IPRs for an anti-scarce resource:

Definition 4.3.3. In case of an anti-scarce resource, the rights $\mathbf{h}$ that should be claimed are

$$\begin{aligned}&\mathbf{h}: X \longrightarrow \hat{Y}_i^{\mathbf{h}}\\ &= \left\{\left\{\hat{Y}_i^{tc} \cup \{\hat{y} \notin \hat{Y}_i^{tc} \mid (\exists\, y_i^{tc} \mid \varepsilon > 0)\}\right\} \setminus \left\{\hat{y} \mid \pi_i(Y^+) \geq \pi_i(\hat{y}), \hat{y} \rightarrow Y^+\right\}\right\}.\end{aligned}$$

Given that some rights are non-separable and that it may sometimes be rationale not to claim all rights in a non- and anti-scarce resource, both CSS and OSS licenses are rational. The above definition of optimal defined IPRs for non- and anti-scarce applications mus therefore be qualified by the caveat that some rights must remain bundled. Based on this, it may be rational either to exclusively retain such a bundle of rights, or to forgo the exclusive claim by choosing OSS. This corresponds to a decision between two second-best solutions since the first-best allocation (see Section 4.2.3) cannot be realized.

Thus, OSS is a rational choice, where the lost of exclusive rights either causes no real costs or the (expected) benefits[17] from anti-rival applications exceeds real costs. Such benefits are more likely where

- the set $\hat{Y}_i^{tc}$ is small, and motives like reputation (Lerner and Tirole, 2002), hobby reasons etc. play a substantial role.
- the number of agents is large so that their code use yields significant cumulative effects.
- agents are able to realize profits by selling goods that are complementary to the software.
- most of the agents are no direct competitors, so that $\exists\, y_i^{tc} \in Y_i^{tc} \mid \varepsilon > 0$ is less likely.

4.3.3 Two Solutions: OSS and CSS

Because ex-post transaction costs lead to non-separable rights, individuals cannot reach first-best IPR-allocation. We have seen that OSS and CSS licensees are two pragmatic solutions to this problem. The reasons not to claim all rights, and—based on this—the individual rationality for OSS or CSS were explained. Based on the analysis above, we now discuss the assets and drawbacks of the two different institutional arrangements that arise from OSS and CSS principles, taking social welfare aspects into account.

[17] We do not discuss how, why, and when 'coordination' (thus contribution) is a equilibrium strategy in such OSS-games. For the purpose of this chapter it is sufficient to note that OSS actually does exist.

4.3.3.1 The Principle of CSS

In case of CSS, only separable rights are traded. As these rights are sold in markets the effects of any applications covered by this rights are in principle internalized.

Of course, also CSS code is used by more than one individual having access to the source code. Here, the solution to the 'dilution-of-control'-problem is to add a governance structure that ensures control based on exclusive ownership, i.e. build a 'firm'. This means that the CSS business strategy is based on maximizing exclusive rights. Firm governance structure permits software creation through the coordinated effort of employed developers. This makes direct control possible so that the required input resources—mainly human capital—are used in an efficient way.

Naturally, there are some hierarchy costs. Because developing software sometimes involves finding solutions developer effort may not be directly observable from output for a problem, one may not be able to conclude directly from the output the effort of the developer. Instead software must be developed in an principal-agent-structure where the principal defines certain aims and assembles a programmer team, but can only imperfectly observe the effort and/or performance of these agents, because of monitoring costs (von Engelhardt and Pasche, 2004, p 9). These and other hierarchy costs limit the size of a firm. For this reason, firms cannot include everybody who could potentially contribute. More specifically, it is impossible to write employment contracts with everyone who could theoretically contribute cumulative activities. This means that CSS firms cannot acquire all available human capital because search, bargaining and enforcement costs are too high. Other kinds of contract based relationships are similarly limited (see p 83) Thus, CSS firms canot reach the theoretically optimal number of agents who *could generate positive effects*.

Of course, positive effects are internalized within a CSS firm. The firm owner can write employment contracts to ensure that he owns not only X but also X^{new}.[18] Furthermore the benefits of knowledge spillovers can also be internalized so long as the employee work for the firm.

[18] Such employment contracts contain a paragraph to ensure that all the possible copyrights are transferred to the company. Thus at the end of the day the employees do not own any IPRs concerning the source code.

However, the set of beneficiaries of the positive effects that require access to the source code remains suboptimally small. Because CSS is only sold in machine-readable forms (binary code), no one outside the firm has access to the source code. This is different from the way copyright protection (and IPR in general) usually works. Normally, IPRs balance ex-ante incentives to produce by ex-post disclosure of the information (Cowan and Harison, 2001; Quah, 2003, pp 16 f, 19 ff).[19] In case of software, the problem of non-separable rights makes this impossible.

4.3.3.2 The Principle of OSS

OSS licenses are non-specific cooperation contracts designed to attract a large number of club members, who can then realize benefits from knowledge spillover and cumulative feedback effects. Since OSS licenses are not limited in time,this cooperation has potentially infinite duration. Anyone can access and use the source code at any time provided that possible legal constraints (e.g. see the GPL) are considered.

This implies that OSS maximizes the number of individuals who can create positive effects. OSS institutions reinforce and to some extend even guarantee these cumulative effects. For example licenses like the GPL force agents to contribute improvements back to the community. Similarly hacker ethics and community norms that support reciprocal behavior also foster cumulative effects. Breaking these rules can lead to informal sanctions in which community members stop cooperation or migrate to other projects (Osterloh et al., 2001, p 16 f). OSS's ability to involve very large numbers of participants is, however at least partially offset by underpro-

[19] Intellectual property law is defined such that it protects the tradeable, hence private internalizeable effects, but forgoes the rights for the not internalizeable effects. Patents do not protect the idea itself but its application in form of machine, method or matter (Besen and Raskind, 1991, p 12). The right to be a temporary monopolist of a novel technical solution is bundled with the constraint to disclosure the information that stands behind it: the technological solution has to be described in the patent specification. Similarly, copyright does not protect the idea itself—the pure information—but its expression. Thus, with e.g. a copyright protected book the author earns money from its publication which is a disclosure of the ideas (or: information). With increasing sales figures, the author earns more money and the ideas of—the information within—that book diffuse.

vision since the benefits of participating (the positive effects) are insufficiently internalized.

OSS licenses also maximize the number of potential beneficiaries, as they permit everyone to use the source code. Of course, restricted licenses like the GPL limit this since CSS producers cannot use GPL-protected code as inputs. However, CSS producers can still read GPL-protected code and learn from it.

OSS licenses transfer the complete set of rights to everyone. This allocation of IPRs makes it impossible to divide by rights of the code consumers from producers, or software developers from coordinators. Instead everyone holds the complete set of rights of the source code.[20] OSS projects are clear structured, they are typically organized according to an 'onion layer' model (Jensen and Scacchi, 2007; Crowston et al., 2006; Wendel de Joode et al., 2003, pp 18,19). The first, outmost layer consists of passive users. In the second layer consists of active users who write bug-reports, test new pre-releases. Layer by layer, the degree of involvement and responsibility increases, see also Figure 1.1, p 13. Community rules govern how participants can move from the outermost layer into the core of the project based on their proven skills and reliability. At the core, so-called *core developers* oversee the design and evolution of the project. The core developers have certain privileges that enable them to control the evolution of the project. This control is enforced through *passive control rights*, i.e. the exclusive right to decide whether or not to accept contributions (McGowan, 2001; Wendel de Joode et al., 2003, p 20). These passive control rights are enforced through exclusive ownership of the database in which the software is stored and legal control of the project's trademarked name.[21] This prevents cloning of projects and protects the signaling function of the project's name.

Despite these pasive control rights and further governance mechanism (see de Laat, 2007) it is reasonable to think that OSS projects have higher coordination costs than CSS firms. Openness and in particular, inclusive

[20] Restricted OSS licensees limit the alienation right for everyone, so all hold the same amount of limited alienation right.

[21] For example, Linux is a trademark of Linus Torvalds, Apache is a trademark of The Apache Software Foundation, KDE and K Desktop Environment logo are registered trademarks of KDE e.V.

licenses can lead to coordination problems like forking[22] or failure to agree on standards. Additionally, OSS governance principles emphasize consensus (Brand and Schmid, 2005), which creates a danger of 'never ending' discussions and other consensus finding costs.

4.3.3.3 The Coexistence of OSS and CSS

OSS and CSS have complementary strength and drawbacks: OSS has weaknesses where CSS has strengths, and vice versa. This is true on both the individual and social levels.

Thus, the coexistence of OSS and CSS can be explained by the fact that different individuals with different sets of tradeable rights and resources need different institutions. Furthermore, the coexistence of OSS and CSS also promote social welfare. On the one hand, CSS is better in using the acquired resources efficiently namely via direct control internalizing the positive effects. This makes CSS a particularly efficient vehicle for developing user-friendly innovations e.g. plug-and-play, easy installation routines and 'nice' graphical user interfaces. CSS also has important advantages for radical innovations because it can internalize benefits from paradigm changes. On the other hand, OSS integrates more human capital and creates spillovers for more individuals. This gives it significant competitive advantage for developing incremental technical and user innovations (von Hippel and von Krogh, 2003; von Hippel, 2005).

These effects could complement each other. For this case, coexisting OSS and CSS institutions would be welfare optimal, see for example von Engelhardt and Swaminathan (2008). The conditions under which this 'optimal' OSS-CSS mix occurs is beyond the scope of this chapter.

4.4 Summary and Outlook

Based on the example of software, the present chapter examines the rationale for OS and CS from a property rights perspective. Our findings can be summarized as follows:

[22] Forking occurs when projects split into multiple incompatible projects because of participants' divergent goals.

1.) The economic characteristics of software, i.e. a source code, make it a non-scarce and anti-scarce resource. Absent transaction costs it is possible to define and trade licenses that optimally allocate IPRs of non- and anti-rival applications.

2.) Some IPRs are non-separable because of ex-post transaction costs. In particular, IPRs that require access to source code cannot be unbundled. This means that the first-best allocation of IPRs that would yield an optimal source code usage cannot be realized. For this reason a first-best realization of contracts is not feasible.

3.) Because of transaction costs, the set of individuals who can contract with one another without a de facto dilution of ownership (i.e. the 'club of source code users') is limited. This limits detailed contracting. Non-specific licenses have lower transaction costs and can reach more individuals.

4.) It is rational to forgo some rights in a non- and anti-scarce resource where (a) no real costs are incurred, (b) feedback mechanisms deliver benefits that exceed the real costs, or (c) both. The net-benefits (payoff) from feedback mechanisms are ceteris paribus higher where more profits can be earned by selling goods and services that are complementary to the software. Of course, this is true for non-separable rights as well.

5.) Rights that cannot be separated must stay are bundled. Economic agents must therefore choose whether or not exclusively assert ownership over the entire bundle or not. CSS only trades the separable rights. Here, production takes place in firms based on exclusive ownership of the source code. OSS minimizes these ownership restrictions in favor of unspecific contracts that permit the transfer of bundled rights.

6.) Based on this we can derive a more *general* statement regarding OS and CS principles. OS and CS principles are likely to occur and be successful if the following holds. There exist an anti-scarce resource[23]

[23] Non-scarcity is of less importance here. The crucial point for the dynamics of OSS development are the feedback-effects, the cumulative effects.

where some property rights covering anti-scarce applications are non-separable. Here agents have to choose whether to keep the entire bundled rights (CS principle) or forgo them (OS principle). In case of the latter, control mechanisms can be based on exclusively hold trademarks and community norms.

Both OS and CS are rational choices. Agents will usually decide based on the individual costs and benefits of each alternative depending on the individual's set of tradeable applications/rights, expected feedback effects, individual resources and the ability to earn profits from complementary products.

7.) From a social welfare perspective OS and CS (software) are both second-best arrangements each of which offers specific benefits and drawbacks. The CS principle benefits from direct (monetary) incentives and control, but has limited scope (size) because of transaction costs. OSS benefits from spillovers and takes advantage of human capital that CSS firms cannot acquire. However, this same openness can also lead to coordination costs (consensus finding), free riding, underprovision, and forking. Since both IPR regimes are imperfect, the coexistence of OS and CS can be welfare optimal.

The analysis presented in this chapter also suggests some further research questions: How do the 'passive control rights' in detail work and are they efficient? This might be analyzed with a formal model of passive control. The rationale for OS and CS principles is developed based on the example of software. What can this framework tell about e.g. the case of OS biotechnology? Under which conditions is an OS-CS mix welfare optimal? Will such optimal mixes occur, and/or be stable? How can firms' decisions between OS- and CS-based business models be analyzed in a simple way? The next two chapters deal with some of these questions. We will focus on the strategic nature of OS vs. CS business models and analyze welfare of industries with OS and CS firms.

5 Quality Competition or Quality Cooperation?

5.1 Introduction*

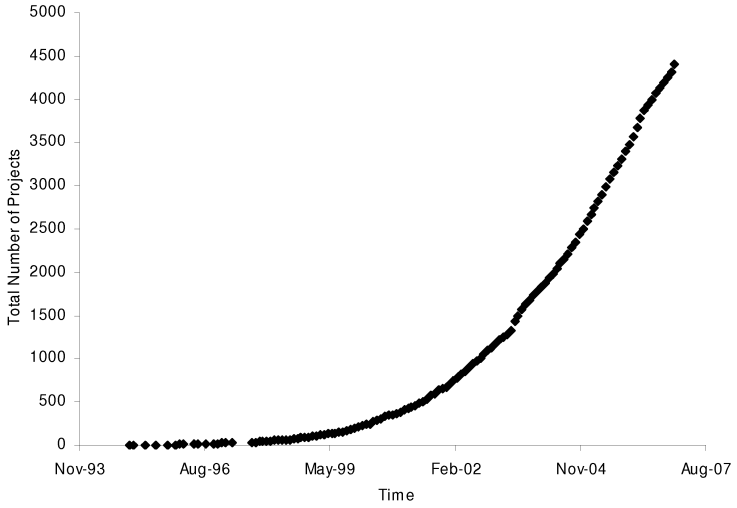
Over the last years, more and more firms have begun to use OS-based business models and develop OS software. In a recent article, Deshpande and Riehle (2008) use automated web searches to compile a worldwide census of OS projects (Figure 5.1). Within their representative sample they report exponential growth from about 500 projects in 2001 to 4,500 in 2007. Strikingly, much of this growth appears to be commercially motivated, i.e. depends on substantial backing from corporations that expect to see a dollars-and-cents return on their investment. Within this category, the great majority of projects involve a shared code base that no single company owns or controls.[1] Each company then extracts benefits by selling complementary products that use this shared code. In practice, these products are very diverse and can include physical hardware (e.g. servers, cell phones), software (applications programs) and services (education, customization services) (Riehle, personal communication).

This chapter focus on the incentives of profit-oriented firms developing OS vs. CS code. Unlike traditional joint venture partners, OS collaborators have no formal obligation to contribute any particular level of effort to these projects. Instead, companies must continuously balance the cost-savings from shared code development against the risk that they will make their competitors' products more desirable. This is true not just for OSS but for all of OS business models. We explore the first general differentiated

*This Chapter is based on von Engelhardt (2010)

[1] Survey research has similarly detected a secular shift in OS motivations away from hobbyist-based production to a culture dominated by paid professionals working on company time (Ghosh et al., 2002a,b).

Figure 5.1: Total Number of Open Source Projects



(Source: Deshpande and Riehle, 2008)

Cournot model in which firms invest in shared OS or private CS code to increase the quality (and profitability) of bundles containing a proprietary good. We discuss the role of OS license-type and the strategic nature of OS vs. CS business models

5.2 Previous Research and the Contribution of this Chapter

Contrary to the rich empirical research,[2] there is less theoretical work on OS vs. CS business models, mostly limited to duopoly cases:

Baake and Wichmann (2004) analyze a duopoly-model where firms that can publish parts of their software as OS. Publishing code leads to positive spillovers i.e. reduce the firms' coding costs, but induce higher coding expenditures and thus decreases the firms' profits if their programs are sub-

[2] See the literature overview in Section 1.3.4.

stitutes. Additionally, it encourages entry and increases the expenditures required to deter entry. They find that both firms invest in OS to increase the quality and profitability of their respective CS products. However, each company must also increase the quality of its CS product to retain its customers. This effect is even stronger when the CS products compete with OS code and/or companies deliberately keep CS quality (and development costs) high to deter entry. Though intriguing, these results are limited to the duopoly case. Baake and Wichmann also make the very special assumption that OS costs rise faster than CS costs. Verani (2006) presents a Bertrand-duopoly model in which companies invest in either OS or CS software and then build products that use it. She finds that firms invest more when their products are substitutes, and that this effort is greater when OS software is used. Schmidtke (2006) analyzes OS business models in a non-differentiated Cournot oligopoly. Firms produce a homogeneous private good (e.g. a computer server) and invest in the quality of a homogeneous public good (OS software). He finds that increasing the number of firms in the market promotes welfare. The effects on each firm's output, prices, and profits depend on the slope of the marginal costs of software development.

Henkel's 2006 "Jukebox Mode of Innovation" uses a duopoly model to explore the case of embedded Linux. Crucially, Henkel assumes that all technologies are developed in-house without shared production of any kind; firms can, however, share costs by disclosing completed technologies to one another. Given this set-up, Henkel finds that each OS firm concentrates on developing whichever technology is most valuable to its business and copies the other technology from its rival. This creates a dynamic in which each company specializes in and controls the technology it values most so that total industry technology spending is biased upward. Henkel finds that OS industries deliver more technology and higher profits provided that firms do not compete too strongly with one another. However, these advantages disappear where both firms' products receive the same quality-increment from each technology. In this case, OS firms are reluctant to make their competitors stronger and therefore invest less than CS firms. Furthermore, firms are most likely to choose OS business models when competition is low, and each firm's technology needs are different.

Finally, two recent papers, Llanes and de Elejalde (2009) and Casadesus-Masanell and Llanes (2009), provide models in the tradition of Hotelling's model. Both models feature a continuum of consumers value the available products diferent, i.e. have heterogeneous tastes. Furthermore, each consumer buys just one package (bundle) or nothing. In Casadesus-Masanell and Llanes (2009) consumers consume software and a complementary service. The software is further segmented into a core program which consumers can use as a free-standing unit, and extensions which are valueless without the core unit. They then examine how firms decide whether to develop one or both software components as OS or CS. Three cases are considered here: a monopoly, a firm vs. non-profit OS project, and duopoly. They find, inter alia, that firms are more willing to open modules when (a) consumer demand for the complementary good is strong, and (b) the quality of OS software is boosted by exogenous user innovation at no cost. Llanes and de Elejalde (2009) similarly consider a model in which each firm sells packages consisting of a primary good (which can be OS or CS) and a complementary private good. Consumers have idiosyncratic preferences so that they usually favor one firm's private good over others. However, rival firms can overcome this preference by investing in a technology that simultaneously increases the quality of both the primary good and also the complement. Llanes and DeElejalde present a two stage model in which a predetermined number of firms (a) decide whether to produce OS or CS in the primary good, and then (b) simultaneously decide the quality/price of the bundle thats they will offer to consumers. They find that when most of the bundle's value comes from the primary good OS firms find it hard to appropriate profits from their investment in an open complement. This leads to outcomes in which a small number of firms choose CS and capture most of the market by delivering high quality code; the other firms become OS and deliver comparatively low quality code at a low price. However, this situation changes where consumers value the complement roughly as much as the primary. In this case, the cost advantage of code-sharing dominates so that all firms choose to become OS even though a hypothetical CS firm would produce higher quality software. This (theoretical) CS quality advantage reflects OS firms' limited ability to recover quality gains from consumers. The advantage disappears in cases where most of the bundle's value comes from the complementary good.

Despite these contributions much remains to be done. In particular, there is still no simple but general theoretical analysis of the necessary and sufficient conditions for OS production by profit seeking firms, and for the coexistence of firms with CS- versus OS-based business models (mixed industry). The model of this chapter aims to fill this gap. We provide a general analysis of the economics of OS vs. CS business models. This analysis includes strategic aspects, the role of OS license types, impacts of the non-commercial community (e.g. the hobbyists), and industry equilibria. Because we use a general oligopoly model, we are able to adress a broad range of possible situations from duopoly with completely separated markets to perfect competition (infinite number of firms, perfect substitutes). Although the model is inspired by, and refers to, software, it analyzes the 'economics of commercial open source' in a general way, and can be readily applied to other examples like 'OS biology'.

As already mentioned, OS business models combine commonly developed code with individually produced and sold complements. For this reason, the underlying strategic logic corresponds to a R&D cooperation *without* explicit contracts. In order to analyze the economics of such contract-free collaborations, we develop an oligopoly model where firms can do OS, or CS, or both. This places our model in tradition of (non-)cooperative R&D models, most prominently represented by the work of d'Aspremont and Jacquemin (1988). But different to the cases analyzed by research on (non)-cooperative R&D (e.g. see de Bondt (1997) for an overview) firms' choices between OS and CS determine whether 'spillovers' exists or not. However, note that regardless of whether CS or OS (or a mix) is chosen, firms' actions are always non-cooperative.

Contract-free OS collaborations can include firms from different markets. For example, several firms use Linux for many different products (see Section 1.2.3). It is therefore important to build a model that allows for horizontal product differentiation. Our the model is thus based on the theory of differentiated oligopoly/duopoly proposed by Dixit (1979) and further developed e.g. by Singh and Vives (1984) and Häckner (2000).

We will also analyze the impact of an important institution on the game: the OS license. Different OS projects use different types of licenses (see

also Section 4.3.2.2). For example, so-called public licenses, like the BSD[3] license, do not restrict the use of the software and the source code in any way and thus allows developers to use OS code as input for CS code development. We will call these *liberal licenses*. Other licenses are more restrictive. One famous example is the GPL[4]. This license claims that any further developed software as well any derived work must also be licensed as a whole under the same license. This type of clause is designed to ensure that OS code stays 'open'. We will call licenses that prohibit OS-CS mixed code *restricted licenses*.

The remainder of this chapter proceeds as follows. Section 5.3 introduces the basic model. Since the model is a two-stage game we will solve it by backward induction Section 5.4 solves stage two and Section 5.5 analyzes firms' stage one OS and CS decisions for both liberal and restricted OS licenses. Section 5.6 focus on mixed industries. We calculate the equilibrium ratios of CS and OS firms assuming free entry and exit. Section 5.7 summarizes our findings and provides directions for further research.

5.3 Firms and Closed Source vs. Open Source Business Models: The Basic Model Setup

In many markets, software is sold *bundled* with complementary products like service (maintenance, individualizing) or hardware. Here firms make money with business models that are based on CS or OS, mixed CS-OS code. The basic principle of open source business models is therefore to develop OS code together with other firms or community members, and then earn revenue with selling the (bundled) complements. This means combining a public good and a private good. It also means combining non-cooperative R&D (OS firms do not have an explicit contract with each other) with oligopolistic competition where products can be vertically differentiated (quality) and horizontally differentiated.

Consider, therefore, a market with $n \geq 2$ firms. One arbitrary firm is denoted by i, with $i \in N = \{1, 2, \ldots, n\}$. Each firm $i \in N$ produces some

[3] BSD stands for Berkeley Software Distribution

[4] The GPL (GNU General Public License) is the most popular OS license.

quantity of a horizontal differentiated product, and develops complementary software. Depending on the OS license this software is OS or CS code, or, in case of liberal licenses, an OS-CS mix. The software and product are then sold as bundle q_i. Firm i's software determines the quality of its bundle q_i, i.e. differences in software lead to vertical product differentiation. We therefore need a *utility function* that enables us to take into account horizontal and vertical product differentiation. There are two common approaches in the literature. One version proposed by Sutton (1997, p 618) and used e.g. by Symeonidis (2003) or Deroian and Gannon (2006) takes the form:

$$V = \sum_{i=1}^{n} \left(q_i - \frac{q_i^2}{v_i^2} \right) - 2\gamma \sum_i \sum_{j<i} \frac{q_i}{v_i} \frac{q_j}{v_j} + I. \tag{5.1}$$

A second version introduced by Dixit (1979) for the duopoly case and generalized by Häckner (2000) and Hsu and Wang (2005) to the oligopoly case takes the form:

$$U = \sum_{i=1}^{n} q_i \alpha_i - \frac{1}{2} \left(\sum_{i=1}^{n} q_i^2 + 2\gamma \sum_{i \neq j} q_i q_j \right) + I. \tag{5.2}$$

In both cases, I is the composite good with its price normalized to one. The parameter $\gamma \in [0,1]$ is an inverse measure of the horizontal product differentiation i.e., γ indicates the (horizontal) substitutability between the different products with $\gamma = 1$ for perfect substitutes. The parameters α_i and v_i each represent the quality level of the bundle, such that vertical product differentiation is expressed by different αs and vs. In practice both types of utility function lead to similar results in our model.[5] We will use the *second* one, i.e. function (5.2) in what follows.

We consider a two-stage game that combines competition in quantities with quality-competition/cooperation. This means that firms make separable decisions about both quantity and quality (via software). As we will show, the decision to develop OS instead of CS code amounts to a decision to cooperate on quality instead of competition.

[5] See p 103 including the footnote 8.

1.) In *stage one*, firms decide whether and how much to invest in software development. Hence, they choose their optimal amount of OS or CS. Since software determines quality, stage one represents a *quality decision*. We analyze stage one under both *liberal* and *restricted* licenses.

2.) In *stage two* oligopolistic competition takes place. The firms produce their 'stage two'-products,[6] bundle it with their 'stage one'-software, and compete à la Cournot. This means, in stage two profit-maximizing *quantities* are defined.

The game will be solved by backward induction. We start in the next section by solving stage two. We then analyze stage one in Section 5.5.

5.4 Stage Two: Quantity Decisions

Stage two features oligopolistic quantity competition between firms, where product-bundles are horizontally and vertically differentiated. Without loss of generality, we normalize the marginal cost of the 'stage two'-product to zero. Fixed costs of 'stage two'-products are given by C.[7] Firms seek $\max_{q_i} \pi_i = p_i q_i - c_i - C$, where $p_i = \alpha_i - q_i - \gamma \sum_{j \neq i} q_j$ is the inverse demand function derived from the utility function (5.2), and c_i are the software development costs incurred in stage one.

The resulting equilibrium prices and quantities of this differentiated oligopoly are given by:

$$p_i = q_i = \frac{\alpha_i + \frac{\gamma}{(2-\gamma)} \sum_{j \neq i} \left(\alpha_i - \alpha_j\right)}{2 + \gamma(n-1)} = \frac{\alpha_i + \theta \sum_{j \neq i} \left(\alpha_i - \alpha_j\right)}{h}. \qquad (5.3)$$

[6] Note that the 'stage two'-product *can* be non-shared software. The bundle then consists of 'stage one'-software plus 'stage two'-software. For example a 'Premium Version' of the 'stage one'-software. For simplicity we will reserve the word 'software' for stage one code whether or not the stage two product also consists of software.

[7] If the 'stage two'-product is also software, C represents the software development costs, i.e. the so-called first copy costs of this 'stage-two'-software. See also footnote 6, p 102

The revenue function is simply the square of (5.3) and given by

$$\frac{\left(\alpha_i + \theta \sum_{j \neq i} \left(\alpha_i - \alpha_j\right)\right)^2}{h^2}.$$

This revenue function closely resembles the revenue function we would achieve by using the utility function V (function (5.1), p 101).[8]

We have introduced the variables θ and h because they will be convenient for interpreting our results later on. As already mentioned the model combines quantity competition (Cournot) with quality competition/cooperation. The measure $h = 2 + \gamma(n-1)$ indicates the degree of *quantity competition*, and depends on the number of competitors weighted by the degree of substitution. (The expression $2 + \gamma(n-1)$ is the typical denominator of differentiated Cournot model.) Second, $\theta = \gamma/(2-\gamma)$ indicates how much differences in quality affect each firm's revenue. For this reason, θ indicates degree of *quality competition* and hence measures the incentive to compete rather than to cooperate on quality. Together h and θ allow us to separates the effects of a change of γ. For example, $d\gamma > 0$ has (a) a negative impact on revenues, since a firm's revenue ceteris paribus decrease with an increase in h, and (b) a positive impact on a firm's revenue provided that the firm has an advantage in quality.

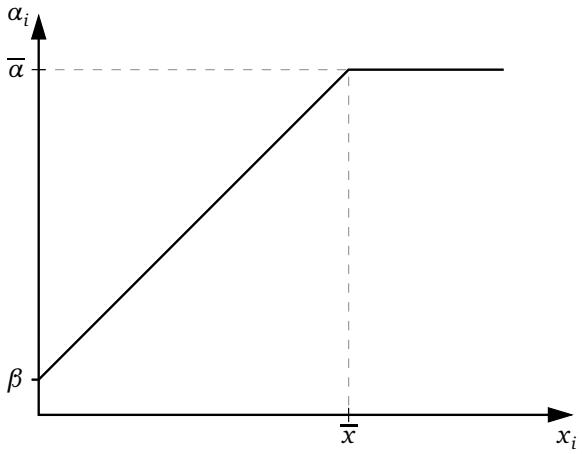
5.5 Stage One: Quality Decisions

Firm i's 'stage one'-software has a direct impact on the quality of the bundle, i.e. on α_i. Let $\alpha_i = \beta + x_i$. The parameter $\beta > 0$ captures the 'stage two'-product's quality (e.g. the quality of the hardware of a mobile phone).

[8] On p 633 Deroian and Gannon (2006) derive a revenue function based on the utility function (5.1) given by $2S\left(u_i(\sigma(n-2)+4) - \sigma\sum_{j\neq i} u_j\right)^2 (4-\sigma)^{-2}(\sigma(n-1)+4)^{-2}$, in their notation. Considering this expression our notation requires us to replace σ by 2γ and replace u_i by α_i. The result only differs from the revenue function of our model by the term $\frac{1}{2}S$. Deroian & Gannon denote with S the number of consumers. Since S represents the size of the market the two expressions differ solely in the *level* of returns. (If we normalize $S = 2$, the two revenue functions become equal.) The fact that the two utility functions lead to similar outcomes has been noted by for example Symeonidis (2003, p 42, Appendix A).

Furthermore, x_i is the software firm i can use for its bundle. For the sake of simplicity, the impact of 'stage one'-software (x_i) on α_i is modeled linearly with an upper boundary $\alpha_i \in [0, \bar{\alpha}]$ that yields a cutoff $\bar{x}$ (Figure 5.2). This feature is needed to ensure that software development cannot shift α_i, and hence demand, to infinity.[9]

Figure 5.2: Impact of Software x_i on Quality



In line with the literature, we assume that software development faces increasing marginal costs as the code base becomes more complex. Since software is in some sense a 'logical machine', and the more sophisticated a software program becomes, the more complex the whole system gets. Modern software development is far more than just writing lines of code, and also consists of finding and fixing mistakes (bugs). Hence the costs of software development consists not only of code-writing but also of bug-avoiding (ex-post designing, coordination and control), bug-finding and bug-fixing costs. Since complexity of modern software rises non-linearly we expect marginal costs to increase (see also von Engelhardt, 2008a, p 14 ff).

[9] The quality function we use can be interpreted as an approximation to a logistic function. Since we assume that firms never develop more that $\bar{x}$, the quality function can also be interpreted as an approximation to an inverse U-shaped function.

We therefore use a function that reflects the conventional computer science wisdom (Brooks' Law) that software costs scale quadratically (Brooks, 1982). Let $\phi > 0$ denote the slope of the marginal costs, and x denote the software code,[10] then the total costs are given by $c(x) = \frac{1}{2} \cdot \phi \cdot x^2$. The total costs are independent of whether the code is OS or CS. (Clearly, OS firms share the total costs.) Thus, OS has no inherent cost advantage over CS. This assumption seems justified given scholars' rudimentary knowledge of this subject (Koch, 2004; Asundi, 2005).[11] A firm that develops CS code bears the total costs of x_i^{cs}, i.e. the firm's costs are $c_i(x_i^{cs}) = \frac{1}{2}\phi x_i^{cs2}$. A Firms that develops OS code is able to share costs with other OS developing firms or members of the community. Depending on its own contribution (x_i^{os}), the firm bears only a fraction k of the total OS costs. Let X^{os} denote the total OS code. Since the firm bears only that fraction that is caused by its own development x_i^{os}, i.e. $k = x_i^{os}/X^{os}$, its costs are $c_i(x_i^{os}) = k \cdot c(X^{os}) = \frac{1}{2}\phi x_i^{os} X^{os}$. All other costs, $(1-k) \cdot c(X^{os})$, are borne by the other OS developing firms and members of the community. Furthermore, this latter holds independently whether the firm develops *only* OS or *mixes* OS and CS code. This implies that the cost function of a firm i that develop OS and CS code is given by $c_i = \frac{1}{2}\phi \left(x_i^{os} + x_i^{cs}\right)^2 + \frac{1}{2}\phi X_{-i}^{os} \left(x_i^{os} + 2x_i^{cs}\right)$, with $X_{-i}^{os} = X^{os} - x_i^{os}$.

In summary code development has both benefits (positive impact on α_i, and hence revenues) and costs. Clearly, a firm's decision to develop OS rather than CS code will affect these costs and benefits. The OS method is a collaborative way of developing *and* using coded, digital goods. While the principle of CS is based on private costs and benefits, the OS principle implies sharing of both costs and benefits:

- Costs: OS code is *jointly developed* code. This implies cost sharing, including complexity costs. (It follows that the marginal costs of OS development are smaller than the marginal costs of CS.)

- Benefits: OS code is *jointly used*. This implies benefit sharing since all firms who use the OS code benefit from its impact on quality. Choos-

[10] Note that this notation captures qualitative and quantitative aspects of the software. Thus, a higher value of x indicates 'more' and 'better' functions.

[11] Some commentators have argued that OS is inherently cheaper to develop than CS software (Raymond, 1998). The current model could be readily extended to cover such scenarios if and when they are confirmed.

ing to develop OS code therefor implies a decision to cooperate rather than to compete on quality. For CS code the opposite holds.

These two aspects determine whether OS or CS is more attractive. They are also strongly influenced by an important institution: the type of OS license. We will see that this institutional difference matters, especially in cases where firms are the *only* potential OS contributors so that no (non-commercial) OS community exists.

5.5.1 Liberal vs. Restricted OS Licenses

We distinguish two types of OS licenses. *Liberal licenses* permit to mix OS with CS code. *Restricted licenses* prohibit any mixing of OS and CS code at the level of 'stage one'-software:[12]

- *Liberal licenses* let firms mix OS and CS code. Thus, α_i is given by

$$\alpha_i = \beta + x_i^{cs} + x_i^{os} + X_{-i}^{os} \tag{5.4}$$

  and the costs c_i are

$$c_i = \frac{1}{2}\phi\left(x_i^{os} + x_i^{cs}\right)^2 + \frac{1}{2}\phi X_{-i}^{os}\left(x_i^{os} + 2x_i^{cs}\right). \tag{5.5}$$

- *Restricted licenses* force firms to *choose* between OS and CS. This separates firms into OS firms and CS firms. Denote the number of OS firms by z and the number of CS firms by r so that $r + z = n$ Similarly denote the of firms by capital letters so that $Z \cup R = N$ and $Z \cap R = \emptyset$. This yields

$$\alpha_i = \begin{cases} \beta + x_i^{cs} & \text{if } i \in R \text{ (CS firm)} \\ \beta + x_i^{os} + X_{-i}^{os} & \text{if } i \in Z \text{ (OS firm),} \end{cases} \tag{5.6}$$

  and for the cost function

$$c_i = \begin{cases} \frac{1}{2}\phi x_i^{cs2} & \text{if } i \in R \text{ (CS firm)} \\ \frac{1}{2}\phi x_i^{os}\left(x_i^{os} + X_{-i}^{os}\right) & \text{if } i \in Z \text{ (OS firm).} \end{cases} \tag{5.7}$$

[12] The prohibition to mix affects stage one! Bundling OS software with 'stage two'-CS software is possible.

Firms maximize profits $\pi_i = p_i \cdot q_i - c_i - C$ with respect to x_i^{os} and/or x_i^{cs}. Firms operating under *liberal licenses* decide their optimal investment for both OS *and* CS code. The resulting reaction functions for firm i's OS and CS output are

$$R_i^{cs} = \frac{(1+(n-1)\theta)\left(\beta - \theta X_{-i}^{cs}\right) - \left(\frac{1}{2}\phi h^2 - (1+(n-1)\theta)\right)\left(x_i^{os} + X_{-i}^{os}\right)}{\frac{1}{2}\phi h^2 - (1+(n-1)\theta)^2}, \tag{5.8}$$

$$R_i^{os} = \frac{\beta - \theta X_{-i}^{cs} - \left(\frac{1}{2}\phi h^2 - (1+(n-1)\theta)\right)x_i^{cs} - \left(\frac{1}{4}\phi h^2 - 1\right)X_{-i}^{os}}{\frac{1}{2}\phi h^2 - 1} \tag{5.9}$$

Because of the liberal licenses firm i's OS and CS are substitutes i.e. $\partial R_i^{cs}/\partial x_i^{os} < 0$ and $\partial R_i^{os}/\partial x_i^{cs} < 0$, with $|\partial R_i^{cs}/\partial x_i^{os}| \geq |\partial R_i^{os}/\partial x_i^{cs}|$.

We obtain the following reaction functions for the OS firms and the CS firms operating under *restricted licenses*:

$$R_{i\in R}^{cs} = \frac{(1+(n-1)\theta)\left(\beta - \theta\sum_{j\neq i} x_j^{cs} - \theta z X^{os}\right)}{\frac{1}{2}\phi h^2 - (1+(n-1)\theta)^2}, \tag{5.10}$$

$$R_{i\in Z}^{os} = \frac{(1+r\theta)\left(\beta - \theta r x^{cs}\right) - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)X_{-i}^{os}}{\frac{1}{2}\phi h^2 - (1+r\theta)^2}. \tag{5.11}$$

Regarding the second order conditions (SOCs), the following holds: The SOC for CS development is the same for liberal and restricted licenses and given by $\phi > \phi_{soc}^{cs}$ with $\phi_{soc}^{cs} = 2(1+(n-1)\theta)^2/h^2$. The SOC for OS development is given by $\phi > \phi_{soc}^{os}$ with for liberal licenses $\phi_{soc}^{os} = 2/h^2$, and for restricted licenses $\phi_{soc}^{os} = 2(1+r\theta)^2/h^2$. This implies that $\phi_{soc}^{cs} > \phi_{soc}^{os}$. As usually done, we assume in the following that each of the SOCs is fulfilled. (Otherwise "more code' is always better so hat firms' output would reach the cutoff $\bar{x}$)

5.5.2 The Strategic Nature of CS Decisions and OS Decisions

Bulow et al. (1985) introduced the concept of strategic complements and substitutes. Decisions of players are strategic substitutes if they mutually inhibit one another. Decisions are strategic complements if the reverse is

true. We will see that OS and CS differ from another along this dimension. We will express this with the elasticities. CS decisions are strategic substitutes since optimal CS development always decreases when other firms supply more code: $\mathcal{E}^{cs}_{cs} = {}^{\partial x^{cs}_i}/_{\partial x^{cs}_j} \cdot {}^{x^{cs}_j}/_{x^{cs}_i} < 0$. However, optimal OS reaction to other player's OS output can either be positive or a negative. If the slope of the marginal costs ϕ is below a certain threshold, decisions about OS are decisions in strategic substitutes, i.e. $\mathcal{E}^{os}_{os} = {}^{\partial x^{os}_i}/_{\partial x^{os}_j} \cdot {}^{x^{os}_j}/_{x^{os}_i} > 0$. This also holds for OS code contributed by non-commercial members of the OS community. We will call such contributions 'community-code' and denote it with x^{os}_{nc}, with "nc" for "non-commercial". Of course, $\mathcal{E}^{os}_{nc} = {}^{\partial x^{os}}/_{\partial x^{os}_{nc}} \cdot {}^{x^{os}_{nc}}/_{x^{os}}$ can also be greater than zero. Finally, the positive feedbacks between the OS players can be very strong. In particular, ${}^{\partial R^{os}_i}/_{\partial x^{os}_j} > 1$ leads to a symmetric Nash-equilibrium where the OS developing firms together develop the cutoff $\bar{x}$. These results can be formally stated as follows:

Proposition 5.5.1. *CS decisions are strategic substitutes: $\mathcal{E}^{cs}_{cs} < 0$.*

Proof. Given the second order condition is fulfilled, then it is true that ${}^{\partial R^{cs}_i}/_{\partial x^{cs}_j} < 0$, which also implies $\mathcal{E}^{cs}_{cs} < 0$. □

Proposition 5.5.2. *OS decisions are strategic substitutes ($\mathcal{E}^{os}_{os} < 0$, $\mathcal{E}^{os}_{nc} < 0$) if and only if $\phi > 2 \cdot \phi^{os}_{soc}$, otherwise strategic complements ($\mathcal{E}^{os}_{os} > 0$, $\mathcal{E}^{os}_{nc} > 0$).*

Proof. If the SOC is fulfilled, ${}^{\partial R^{os}_i}/_{\partial x^{os}_j} < 0$ and ${}^{\partial R^{os}_i}/_{\partial x^{os}_{nc}} < 0$ are true only for $\phi > 2\phi^{os}_{soc}$, otherwise is ${}^{\partial R^{os}_i}/_{\partial x^{os}_j} > 0$ and ${}^{\partial R^{os}_i}/_{\partial x^{os}_{nc}} > 0$. Thus $\mathcal{E}^{os}_{os} < 0$, $\mathcal{E}^{os}_{nc} < 0$ for $\phi > 2 \cdot \phi^{os}_{soc}$. Otherwise $\mathcal{E}^{os}_{os} > 0$, $\mathcal{E}^{os}_{nc} > 0$. □

Proposition 5.5.3. *For $\phi < \phi^{os}_{\bar{x}}$ there exists a symmetric Nash-equilibrium of OS development such that $X^{os} = \bar{x}$.*

Proof. The conditon ${}^{\partial R^{os}_i}/_{\partial x^{os}_j} = 1$ implies the boundary $\phi^{os}_{\bar{x}}$ given by (a) $\phi^{os}_{\bar{x}} = {}^{n}/_{(n+1)} \cdot 2\phi^{os}_{soc}$ in the case of liberal licenses, and (b) $\phi^{os}_{\bar{x}} = {}^{z}/_{(z+1)} \cdot 2\phi^{os}_{soc}$ in the case of restricted licenses. This means that ${}^{\partial R^{os}_i}/_{\partial x^{os}_j} > 1$ for all $\phi < \phi^{os}_{\bar{x}}$, i.e. that OS players react positive to each other with a factor greater than one. Intuitively, the positive strategic interplay of OS is so strong that the players push each other to an upper corner solution equilibrium where the total OS code development reaches the cutoff $X^{os} = \bar{x}$. Symmetry implies that each OS firms in this Nash-equilibrium develops the same fraction of $\bar{x}$. See also Figure C.2 in Appendix C □

Proposition 5.5.3 has the following intuition: If $\phi < \phi^{os}_{soc}$ then developing more OS code is always better regardless of what the other OS players do. The reason is the relatively low slope of the marginal costs. For $\phi^{os}_{soc} < \phi < \phi^{os}_{\bar{x}}$ developing more OS code is still better but now because of what the other OS players do. This positive feedback among players creates strong incentives to cooperate and shifts the 'more OS code is always better' region upwards. Analyzing this boundary provides insight into the strategic interaction among OS firms. Depending on the type of license, $\phi^{os}_{\bar{x}}$ is equal $n/(n+1) \cdot 4/h^2$ or $z/(z+1) \cdot 4(1+r\theta)^2/h^2$. Recall that $h = 2 + (n-1)\gamma$ and $\theta = \gamma/(2-\gamma)$ where θ indicates the incentives to compete on quality. If licenses are restricted then the OS firms cooperate on quality among each other but compete on quality with CS firms. Therefore an increase of γ—which implies an increase of θ—has a positive impact on the incentives to develop more OS software through the term $(1 + r\theta)^2$. At the same time, $d\gamma > 0$ implies $dh > 0$ and thus reduces incentives to develop more OS simply because of increased competition. Finally the total number of firms in the industry has an ambiguous impact on incentives to develop OS software. On the one hand an increase in n and z yields $dh > 0$. On the other hand, if more firms develop OS code then then each individual firm has to contribute less to reach any given level of total OS output. This is expressed by $n/(n+1)$, and $z/(z+1)$ respectively.

For $X^{os} = \bar{x}$ our model's results depend on the exogenously given $\bar{x}$. We therefore concentrate on situations where ϕ is high enough to ensure that total OS code does not reach the cutoff in what follows.

5.5.3 OS and CS in Case of Liberal Licenses

In this section we analyze the case of *liberal licenses*. In this context, the community-code plays the following role. Since liberal license permit mixed OS-CS code, code from the non-commercial OS community (x^{os}_{nc}) directly impacts on both OS and CS development by firms. If there is enough community-code, firms do not develop any further software since they receive everything needed "for free". Recall that in case of liberal licenses all firms can develop OS code. Here x^{os}_{nc} also has a second effect. If the firms' markets are sufficiently separated they develop OS code whether or not community-code exists. For high θ however, firms will only produce OS

code if $x_{\text{nc}}^{\text{os}}$ exceeds some threshold As we will see, firms contribute OS code only if $\theta < 1/(n+1)$ for the case of $x_{\text{nc}}^{\text{os}} = 0$. The reason is that for $\theta < 1/(n+1)$ quality competition is low. Since the substitutability (γ) is low, the firms' markets are sufficiently separated so that incentives to compete on quality are weak which makes quality cooperation more attractive. On the other hand, incentives to compete on quality are higher when $\theta > 1/(n+1)$. In this case firms will only participate in shared OS development if volunteers supply enough $x_{\text{nc}}^{\text{os}} > 0$. Given enough community-code the cost-sharing benefits of OS development are strong enough to ensure firm participation when $\theta > 1/(n+1)$.[13] However, the positive effect of $x_{\text{nc}}^{\text{os}}$ on commercial OS is limited. When $\theta > 1/(n-1)$, firms never develop OS code. Thus for large numbers of firms there are virtually no situations where firms produce OS code under liberal licenses, whether or not $x_{\text{nc}}^{\text{os}} > 0$ (see also Section 5.6.1).

These results can be formally stated as follows:

Proposition 5.5.4. *A symmetric Nash-equilibrium exists in which firms only develop CS* ($x^{cs*} > 0$, $x^{os*} = 0$) *if and only if the conditions* $x_{nc}^{os} < \beta(1+(n-1)\theta)/\eta$ *and* $x_{nc}^{os} < \beta 2\theta(n-1)/\eta$ *are fulfilled, where* $\eta = \frac{1}{2}\phi h^2 - (1+(n-1)\theta)$.

Proof. If all firms develop *only* CS code, then $x_i^{\text{os}} = 0 \, \forall i \in N$ and therefore the symmetric solution of the CS reaction functions—see (5.8)—is given by

$$x^{\text{cs}*} = \frac{\beta(1+(n-1)\theta)}{\frac{1}{2}\phi h^2 - (1+(n-1)\theta)} - x_{\text{nc}}^{\text{os}}.$$

Here, $x^{\text{cs}*}$ is greater zero only if $x_{\text{nc}}^{\text{os}} < \beta(1+(n-1)\theta)/\eta$.[14] Moreover, the optimal amount of OS must be zero to ensure that ($x^{\text{os}} = 0$, $x^{\text{cs}} > 0$) is an equilibrium. Setting the symmetric solution obtained from OS reaction function (5.9) with $X_{-i}^{\text{cs}} = (n-1) \cdot x^{\text{cs}*}$ and $x_i^{\text{cs}} = x^{\text{cs}*}$ equal to zero delivers the boundary $x_{\text{nc}}^{\text{os}} = \beta 2\theta(n-1)/\eta$. For values below this boundary optimal OS is zero. Taking these conditions together we conclude that a Nash-equilibrium with $x^{\text{cs}*} > 0$, $x^{\text{os}*} = 0$ only exists for $x_{\text{nc}}^{\text{os}}$ that satisfy $x_{\text{nc}}^{\text{os}} < \beta(1+(n-1)\theta)/\eta$ and $x_{\text{nc}}^{\text{os}} < \beta 2\theta(n-1)/\eta$. □

[13] In some sense the non-commercial community resembles a firm that develops code but does not compete with the other firms. This 'firm' has an 'individual' $\gamma = 1$, and thus $\theta = 0$. Thus this 'firm' depresses the average θ so that it is again below the threshold.

[14] Otherwise firms do not develop CS as there is no incentive to do so because enough code is delivered by the non-commercial community.

Proposition 5.5.5. *A symmetric Nash-equilibrium exists in which firms only develop OS* ($x^{cs*} = 0$, $x^{os*} > 0$) *if and only if* $\beta((n+1)\theta-1)(n-1)/\eta < x_{nc}^{os} < \beta/\left(\frac{1}{4}\phi h^2-1\right)$, *where* $\eta = \frac{1}{2}\phi h^2 - (1+(n-1)\theta)$.

Proof. If all firms develop *only* OS, then $x_i^{cs} = 0\,\forall i \in N$ and the symmetric solution of the OS reaction functions—see (5.9)—is given by

$$x^{os*} = \frac{\beta - \left(\frac{1}{4}\phi h^2 - 1\right) x_{nc}^{os}}{\frac{1}{4}\phi h^2 + n\left(\frac{1}{4}\phi h^2 - 1\right)}.$$

First, x^{os*} has to be greater zero. If $\phi < 2\phi_{soc}^{os}$ (strategic complements) then $x^{cs*} > 0\ \forall x_{nc}^{os}$. Otherwise $x^{os*} > 0$ only if $x_{nc}^{os} < \beta/\left(\frac{1}{4}\phi h^2-1\right)$.[15] Second, the optimal amount of CS must be zero to ensure that $(x^{os} > 0, x^{cs} = 0)$ is an equilibrium. Setting the symmetric solution obtained from the CS reaction function (5.8) with $X^{os}{}_{-i} = (n-1)\cdot x^{os*} + x_{nc}^{os}$ and $x_i^{os} = x^{os*}$ equal to zero delivers the boundary $x_{nc}^{os} = \beta((n+1)\theta-1)(n-1)/\eta$. Taking both conditions together we find that a Nash-equilibrium with $x^{cs*} = 0$, $x^{os*} > 0$ only exists for $\beta((n+1)\theta-1)(n-1)/\eta < x_{nc}^{os} < \beta/\left(\frac{1}{4}\phi h^2-1\right)$. □

Notice that in both cases code-output falls when quantity competition (h) increases. Hence $\partial x^{cs}/\partial h < 0$ and $\partial x^{os}/\partial h < 0$. The reason is that strong quantity competition limits the appropriability of quality investments in any case. Furthermore, while CS output reacts to the degree of quality competition ($\partial x^{cs}/\partial\theta > 0$), the OS-only case does not depend on θ. The reason is that OS firms cooperate on quality: they avoid quality competition through code-sharing. Finally, the OS function also reflects the cost-sharing aspect of OS by $\left(1/4\phi h^2 - 1\right)$.

Proposition 5.5.6. *A symmetric Nash-equilibrium exists in which the firms develop OS and CS* ($x^{cs*} > 0$, $x^{os*} > 0$) *if and only if* $\beta((n+1)\theta-1)(n-1)/\eta < x_{nc}^{os} < \beta 2\theta(n-1)/\eta$, *where* $\eta = \frac{1}{2}\phi h^2 - (1+(n-1)\theta)$.

Proof. Optimal CS and OS output are given by the reaction functions (5.8) and (5.9). Furthermore, symmetry guaranties that $\forall i$: $x_i^{os} = x^{os}$ and

[15] If x_{nc}^{os} exceeds this level, then there is so much community code that firms do not have any incentives to develop any further code.

$x_i^{cs} = x^{cs}$ in equilibrium. Reciprocal substitution then leads to the following solutions for (5.8) and (5.9):

$$x^{cs*} = \frac{\beta\,(1-(n+1)\,\theta)}{\frac{1}{2}\phi h^2 - (1+(n-1)\,\theta)} + \frac{x_{nc}^{os}}{n-1},$$

$$x^{os*} = \frac{\beta 2\theta}{\frac{1}{2}\phi h^2 - (1+(n-1)\,\theta)} - \frac{x_{nc}^{os}}{n-1}.$$

Using the non-negativity constraints $x^{cs*}>0$ and $x^{os*}>0$ we arrive at the two conditions $x_{nc}^{os} > {\beta((n+1)\theta-1)(n-1)}/{\eta}$ and $x_{nc}^{os} < {\beta 2\theta(n-1)}/{\eta}$. □

Proposition 5.5.7. *If $\theta > {1}/{(n-1)}$ any software development by firms is entirely CS.*

Proof. For the case $\theta > {1}/{(n-1)}$ we find that ${\beta((n+1)\theta-1)(n-1)}/{\eta} > {\beta 2\theta(n-1)}/{\eta}$. This means that there is no equilibrium for $x^{cs*} > 0$, $x^{os*} > 0$ (see proposition 5.5.6). Furthermore, $\theta > {1}/{(n-1)}$ implies that ${\beta((n+1)\theta-1)(n-1)}/{\eta} > {\beta}/{\left(\frac{1}{4}\phi h^2-1\right)}$, where $\eta = \frac{1}{2}\phi h^2 - (1+(n-1)\,\theta)$. This yields that an equilibrium with $x^{cs*} = 0$, $x^{os*} > 0$ cannot exist for $\theta > {1}/{(n-1)}$ (see proposition 5.5.5). Finally, $\theta > {1}/{(n-1)}$ implies that ${\beta(1+(n-1)\theta)}/{\eta} < {\beta 2\theta(n-1)}/{\eta}$. This means that an equilibrium exists with $x^{cs*} > 0$, $x^{os*} = 0$ if $x_{nc}^{os} < {\beta(1+(n-1)\theta)}/{\eta}$ since both conditions of proposition 5.5.4 are fulfilled. □

Proposition 5.5.8. *In the case where the non-commercial community supplies zero code ($x_{nc}^{os} = 0$) firms only develop OS if $\theta < {1}/{(n+1)}$.*

Proof. For $x_{nc}^{os} = 0$ the upper conditions of both equilibria with OS, i.e. $x^{cs*} = 0$, $x^{os*} > 0$ and $x^{cs*} > 0$, $x^{os*} > 0$, are met. With $x_{nc}^{os} = 0$ the lower boundary of both types of equilibria with OS are the same (compare 5.5.5 with 5.5.6) and yield the condition $\beta(n-1)((n+1)\,\theta-1) < 0$. This is met if and only if $\theta < {1}/{(n+1)}$. □

The impact of x_{nc}^{os} can be summarized as follows. For CS-only ($x^{cs*} > 0$, $x^{os*} = 0$) and OS-only ($x^{cs*} = 0$, $x^{os*} > 0$) Nash-equilibria the following holds: Except where OS decisions are strategic complements ($\phi < 2\phi_{soc}^{os}$), x_{nc}^{os} crowds out firm-developed code. In both cases firms substitute the community-code x_{nc}^{os} for their own code-output. The impact of $dx_{nc}^{os} > 0$

is different for the CS-OS equilibrium ($x^{cs*} > 0$, $x^{os*} > 0$). Here is both a crowding-out and a crowding-in, since $\partial x^{cs*}/\partial x_{nc}^{os} > 0$ and $\partial x^{os*}/\partial x_{nc}^{os} < 0$ with $\partial x^{cs*}/\partial x_{nc}^{os} = -\partial x^{os*}/\partial x_{nc}^{os}$. Additionally, assumed $1/(n+1) < \theta < 1/(n-1)$, an equilibrium in which firms develop OS code only exists only if $x_{nc}^{os} > \beta((n+1)\theta-1)(n-1)/\eta$. Otherwise, only CS code exists since the incentives to cooperate are too low. For $\theta > 1/(n-1)$ no firms develop OS software at all, regardless of x_{nc}^{os}.

5.5.4 OS and CS in Case of Restricted Licenses

In the case of restricted licenses we must analyze (a) how much software CS firms ($i \in R$) and OS firms ($i \in Z$) develop in equilibrium, and (b) the number of CS and OS firms that coexist in the industry in equilibrium. We will examine the latter in Section 5.6.2. Here we derive x^{cs*} and x^{os*} for industries containing any arbitrary number of CS and OS firms.

The symmetric solution of (5.10) delivers the symmetric Nash-equilibrium regarding the interaction among the CS firms, thus optimal CS output for a given amount of OS:

$$x^{cs} = \frac{(1+(n-1)\theta)\left(\beta - z\theta X^{os}\right)}{\frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+z\theta)}$$

for all $X^{os} < \beta/z\theta$, otherwise $x^{cs} = 0$. Similarly, the symmetric solution for (5.11) leads to the symmetric Nash-equilibrium of the interaction among the OS firms, thus optimal OS for a given amount of CS:

$$x^{os} = \frac{(1+r\theta)\left(\beta - \theta\sum x^{cs}\right) - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)x_{nc}^{os}}{\frac{1}{4}\phi h^2(1+z) - z(1+r\theta)^2}$$

for all $\sum x^{cs} < \beta/\theta - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)x_{nc}^{os}/(1+r\theta)\theta$, otherwise $x^{os} = 0$.[16]

[16] As mentioned above we focus on cases where $\phi > \phi_{\bar{x}}^{os}$. In the case of $\phi < \phi_{\bar{x}}^{os}$ multible equilibria exist, see Appendix C.

Because of symmetry we can replace $X^{\text{os}} = zx^{\text{os}} + x^{\text{os}}_{\text{nc}}$ and $\sum x^{\text{cs}} = rx^{\text{cs}}$. For convenience we also rewrite the denominator of x^{cs} and x^{cs} with ψ and χ. This yields

$$x^{\text{cs}} = \frac{(1+(n-1)\theta)\left(\beta - z\theta\left(zx^{\text{os}} + x^{\text{os}}_{\text{nc}}\right)\right)}{\psi} \tag{5.12}$$

$$x^{\text{os}} = \frac{(1+r\theta)\left(\beta - \theta r x^{\text{cs}}\right) - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)x^{\text{os}}_{\text{nc}}}{\chi} \tag{5.13}$$

As before, we analyze the impact of quantity competition and quality competition on OS and CS output seperately. In the case where no CS firms exist ($r = 0$ and thus $n = z$) OS is again only affected by the degree of quantity competition (h) with $\partial x^{\text{os}}/\partial h < 0$. If the OS firms face CS competitors, however, they must also compete with the CS firms on quality. On the other hand, CS firms compete on quality whether or not OS firms exist. More formally: for $z = 0$ we have $\partial x^{\text{cs}}/\partial\theta > 0$ and $\partial x^{\text{cs}}/\partial h < 0$.

We now analyze the interactions between the CS- and OS firms. It turns out that CS is a strategic substitute to OS and vice versa.

Proposition 5.5.9. *For CS development, OS code is a strategic substitute, i.e.* $\mathcal{E}^{cs}_{os} < 0$ *and* $\mathcal{E}^{cs}_{nc} < 0$.

Proof. $\mathcal{E}^{\text{cs}}_{\text{os}} = \partial x^{\text{cs}}/\partial x^{\text{os}} \cdot x^{\text{cs}}/x^{\text{os}} < 0$ and $\mathcal{E}^{\text{cs}}_{\text{nc}} = \partial x^{\text{cs}}/\partial x^{\text{os}}_{\text{nc}} \cdot x^{\text{os}}_{\text{nc}}/x^{\text{cs}} < 0$. □

Proposition 5.5.10. *For OS development, CS code is a strategic substitute,i.e.* $\mathcal{E}^{os}_{cs} < 0$.

Proof. $\mathcal{E}^{\text{os}}_{\text{cs}} = \partial x^{\text{os}}/\partial x^{\text{cs}} \cdot x^{\text{cs}}/x^{\text{os}} < 0$ for all $\chi > 0$. (The case where $\chi < 0$, i.e. OS firms produce the cutoff is adressed in the Appendix C.) □

Solving (5.12) and (5.13) yield the Nash-equilibria where firms simultaneous choose OS *and* CS for indusries containing any arbitrary number of OS and CS firms. The solutions include equilibria where only OS firms develop software, only CS firms develop software, and where both OS and CS firms develop software. These results can be formally stated as follows:

Proposition 5.5.11. *A Nash-equilibrium exists in which only CS firms develop software* ($x^{cs*} > 0$, $x^{os*} = 0$) *in the case where*

- *OS decisions are strategic complements or relatively weak strategic substitutes ($\mathcal{E}_{nc}^{os} > -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$) if and only if $x_{nc}^{os} < \beta/z\theta$ and also $x_{nc}^{os} < \beta(1+r\theta)\kappa/\mu$, or*
- *OS decisions are relatively strong strategic substitutes ($\mathcal{E}_{nc}^{os} < -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$), if and only if $\beta(1+r\theta)\kappa/\mu < x_{nc}^{os} < \beta/z\theta$,*

where $\kappa = \frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+\theta n)$, *and* $\mu = \frac{1}{2}h^2\phi\left[\frac{1}{4}h^2\phi - (1+r\theta)^2\right] - (1+(n-1)\theta)\left[\frac{1}{4}h^2\phi(1+z\theta) - (1+r\theta)(1+\theta n)\right]$.

Proof. In the case where only CS firms develop software ($x^{os*} = 0$) CS output is given by

$$x^{cs*} = \frac{(1+(n-1)\theta)\left(\beta - z\theta x_{nc}^{os}\right)}{\frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+z\theta)}.$$

Since x^{cs*} must be greater zero, x_{nc}^{os} must satisfy the condition $x_{nc}^{os} < \beta/z\theta$. Second, x^{os*} must be zero. We find from (5.13) that $r \cdot x^{cs*} > \beta/\theta - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)x_{nc}^{os}/(1+r\theta)\theta$. Inserting here the above expression for x^{cs*} and solving for x_{nc}^{os} yields $x_{nc}^{os} < \beta\kappa/\mu$ if $\mu > 0$ and $x_{nc}^{os} > \beta\kappa/\mu$ otherwise. For $\mu > 0$ is $(1+(n-1)\theta)z\theta/\left[\frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+z\theta)\right] > \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right)/(1+r\theta)r\theta$. This expression can be rewritten as $\mathcal{E}_{nc}^{os} > -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$. □

We interpret proposition 5.5.11 is as follows. *First*, there must be CS development. Recall that $\mathcal{E}_{nc}^{cs} < 0$, i.e. OS software from the non-commercial community has a negative impact on CS output. Thus, it must be that $x_{nc}^{os} < \beta/z\theta$, otherwise x_{nc}^{os} completely suppresses code-development by CS firms regardless how much OS code firms produce. *Second*, the CS firms must develop enough CS to completely suppress OS code development by firms. Suppose that $x_{nc}^{os} = 0$. In such a case x^{cs*} supresses commercial OS code if $rx^{cs*} > \beta/\theta$. This condition is fulfilled where the marginal costs of software-development rise relatively slowly, i.e. where $\phi < \phi_{soc}^{cs} \cdot (1+n\theta)/(1+(n-1)\theta)$. Where the marginal costs rise faster, the cost-sharing aspect of OS guarantees that OS firms always develop some code. Suppose now that $x_{nc}^{os} > 0$. Here we have to take into account that x_{nc}^{os} has direct and indirect impact on x^{os}. The direct effect of x_{nc}^{os} on x^{os} is expressed by $\mathcal{E}_{nc}^{os}$ and can be positive or negative. The indirect effect occurs because x_{nc}^{os} affect x^{cs} which in turn determine x^{os}. Here x_{nc}^{os} has a positive impact on x^{os} because x_{nc}^{os}

decreases x^{cs} ($\mathcal{E}^{cs}_{nc} < 0$), and this has a positive impact on x^{os} ($\mathcal{E}^{os}_{cs} < 0$). The overall effect is positive where OS decisions are only weak strategic substitutes or even strategic complements. This means that the amount of x^{os}_{nc} must be bounded for $x^{os} = 0$. But if x^{os}_{nc} strongly crowds out firm-OSS ($\mathcal{E}^{os}_{nc} < -\mathcal{E}^{cs}_{nc}\mathcal{E}^{os}_{cs}$) it suppresses firm-OS. (Where OS decisions are very strong strategic substitutes firm-OS is suppressed even for $x^{os}_{nc} = 0$. The formal condition is: $\beta(1+r\theta)\kappa/\mu < 0$).

Proposition 5.5.12. *A Nash-equilibrium exists in which only OS firms develop software* ($x^{cs*} = 0$, $x^{os*} > 0$) *where*

- *OS decisions are strategic substitutes ($\mathcal{E}^{os}_{nc} < 0$), if and only if $\beta\sigma/\frac{1}{4}h^2\phi\theta < x^{os}_{nc} < \beta(1+r\theta)/\left(\frac{1}{4}h^2\phi-(1+r\theta)^2\right)$, or*
- *OS decisions are strategic complements ($\mathcal{E}^{os}_{nc} > 0$), if and only if $\beta\sigma/\frac{1}{4}h^2\phi\theta < x^{os}_{nc}$,*

with $\sigma = \chi - z^2\theta\,(1+r\theta)$

Proof. If only the OS firms develop software ($x^{cs*} = 0$) then commercial OS output is

$$x^{os*} = \frac{(1+r\theta)\,\beta - \left(\frac{1}{4}\phi h^2 - (1+r\theta)^2\right) x^{os}_{nc}}{\chi}.$$

First, x^{os*} has to be greater zero. If OS decisions are strategic substitutes, non-commercial OS has to satisfy $x^{os}_{nc} < (1+r\theta)\beta/\left(\frac{1}{4}h^2\phi-(1+r\theta)^2\right)$.[17] If OS decisions are strategic complements, $x^{os*} > 0\ \forall x^{os}_{nc}$. Second, since x^{cs*} must be zero, we obtain from (5.12) the condition $zx^{os*} + x^{os}_{nc} > \beta/z\theta$. This finally yields $x^{os}_{nc} > (\chi - z^2\theta(1+r\theta))\beta/\frac{1}{4}h^2\phi\theta$. □

Proposition 5.5.12 is straightforward. Any x^{os}_{nc} fosters OS development by firms if OS decisions are strategic complements. When OS decisions are strategic substitutes, x^{os}_{nc} crowds out firm OS. Furthermore, total OS (X^{os}) must be greater than $\beta/z\theta$ to ensure that the CS firms do not develop code. This condition requires sufficient community-code in cases

[17] If x^{os}_{nc} exceeds this level, then there is so much community-code that the OS firms do not have any incentives to develop own code.

where commercial code output is insufficient. The condition is met without community-code only if the marginal costs of software-development rise relatively slowly. OS firms jointly produce enough code to completely suppress CS if $\phi < {}^{(1+n\theta)}/_{(1+r\theta)} \cdot {}^{z}/_{(z+1)} \cdot 2 \cdot \phi_{\text{soc}}^{\text{os}}$. In this case $\left(\chi - z^2\theta(1+r\theta)\right)\beta/\frac{1}{4}h^2\phi\theta < 0$ and the condition is fulfilled $\forall x_{\text{nc}}^{\text{os}}$.

Proposition 5.5.13. *A Nash-equilibrium exists in which both types of firm develop software* ($x^{cs*} > 0$, $x^{os*} > 0$)

(a) for $\mathcal{E}_{cs}^{os}\mathcal{E}_{os}^{cs} < 1$ *where*

- *OS decisions are strategic complements or relatively weak strategic substitutes* ($\mathcal{E}_{nc}^{os} > -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$)*, if and only if* ${}^{\beta(1+r\theta)\kappa}/_{\mu} < x_{nc}^{os} < {}^{\beta\sigma}/_{\frac{1}{4}h^2\phi\theta}$,
- *OS decisions are relatively strong strategic substitutes* ($\mathcal{E}_{nc}^{os} < -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$)*, if and only if* $x_{nc}^{os} < {}^{\beta\sigma}/_{\frac{1}{4}h^2\phi\theta}$ *and also* $x_{nc}^{os} < {}^{\beta(1+r\theta)\kappa}/_{\mu}$,

(b) for $\mathcal{E}_{cs}^{os}\mathcal{E}_{os}^{cs} > 1$ *where*

- *OS decisions are strategic complements or relatively weak strategic substitutes* ($\mathcal{E}_{nc}^{os} > -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$)*, if and only if* ${}^{\beta\sigma}/_{\frac{1}{4}h^2\phi\theta} < x_{nc}^{os} < {}^{\beta(1+r\theta)\kappa}/_{\mu}$,
- *OS decisions are relatively strong strategic substitutes* ($\mathcal{E}_{nc}^{os} < -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$)*, if and only if* $x_{nc}^{os} > {}^{\beta\sigma}/_{\frac{1}{4}h^2\phi\theta}$ *and also* $x_{nc}^{os} > {}^{\beta(1+r\theta)\kappa}/_{\mu}$,

with $\sigma = \chi - z^2\theta(1+r\theta)$*, and* $\kappa = \frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+\theta n)$*, and* $\mu = \frac{1}{2}h^2\phi\left[\frac{1}{4}h^2\phi - (1+r\theta)^2\right] - (1+(n-1)\theta)\left[\frac{1}{4}h^2\phi(1+z\theta) - (1+r\theta)(1+\theta n)\right]$.

Proof. The simultaneous solution of (5.12) and (5.13) is given by

$$x^{\text{cs}*} = (1+(n-1)\theta)\frac{\beta\sigma - z\theta\frac{1}{4}\phi h^2 x_{\text{nc}}^{\text{os}}}{\psi\chi - (1+\theta r)\theta^2 r(1+(n-1)\theta)z^2}, \tag{5.14}$$

and

$$x^{\text{os}*} = \frac{\kappa\beta(1+\theta r) - \mu x_{\text{nc}}^{\text{os}}}{\psi\chi - (1+\theta r)\theta^2 r z^2(1+(n-1)\theta)}. \tag{5.15}$$

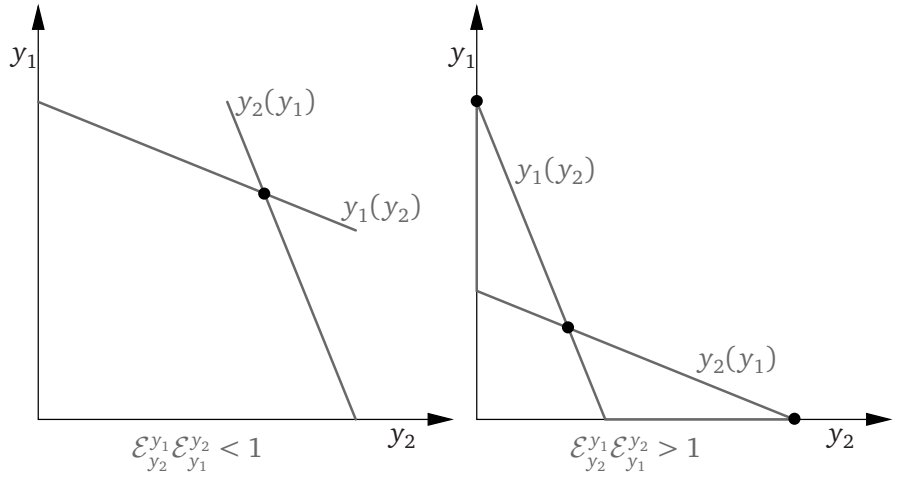
The denominator of both expressions is greater zero if and only if $\mathcal{E}_{\text{cs}}^{\text{os}}\mathcal{E}_{\text{os}}^{\text{cs}} > 1$.[18] First, consider the case where $\mathcal{E}_{\text{cs}}^{\text{os}}\mathcal{E}_{\text{os}}^{\text{cs}} > 1$. If the denominator is

[18] Notice that we have defined the elasticities with respect to (5.12) and (5.13). Thus the symmetry of OS firms, and of CS firms, each was taken into account. Namely the derivatives are ${}^{\partial x^{\text{cs}}}/_{\partial x^{\text{os}}} = -{}^{(1+(n-1)\theta)z^2\theta}/_{\psi}$ and ${}^{\partial x^{\text{os}}}/_{\partial x^{\text{cs}}} = -{}^{(1+r\theta)r\theta}/_{\psi}$.

positive, then x^{cs} and x^{os} are positive if the numerators have positive signs. Then $x^{cs*} > 0$ leads to the condition $x_{nc}^{os} < {}^{\beta\sigma}/_{\frac{1}{4}h^2\phi\theta}$. The condition for $x^{os*} > 0$ depends on whether $\mu > 0$ or not. Where $\mu > 0$ we know that $\mathcal{E}_{nc}^{os} < -\mathcal{E}_{nc}^{cs}\mathcal{E}_{cs}^{os}$ which implies that the condition for $x^{os*} > 0$ is $x_{nc}^{os} < {}^{\beta(1+r\theta)\kappa}/_{\mu}$. Otherwise the opposite holds. Second, consider the case $\mathcal{E}_{cs}^{os}\mathcal{E}_{os}^{cs} < 1$. If the denominator is negative, then x^{cs} and x^{os} are both positive if the numerators have negative signs. This reverses the unequal signs of the conditions. □

The logic behind proposition 5.5.13 is the following. OS and CS are linear functions of each other. Furthermore, OS and CS react on each other as strategic substitutes (see propositions 5.5.9 and 5.5.10). For this reason a Nash-equilibrium exists with ($x^{cs*} > 0$, $x^{os*} > 0$) if and only if either (a) neither ($x^{cs*} > 0$, $x^{os*} = 0$) nor ($x^{cs*} = 0$, $x^{os*} > 0$) exist, or (b) both ($x^{cs*} > 0$, $x^{os*} = 0$) and ($x^{cs*} = 0$, $x^{os*} > 0$) exist. Figure 5.3 illustrates this logic with a symmetric example of decisions in strategic substitutes. The

Figure 5.3: Linear Strategic Substitutes and Equilibrium



left hand side has only one equilibrium, the inner solution $y_1^* > 0, y_2^* > 0$. The right hand side has three equilibria. The inner solution ($y_1^* > 0, y_2^* > 0$) and the two corner solutions $y_1^* > 0, y_2^* = 0$ and $y_1^* = 0, y_2^* > 0$. This implies that a Nash-equilibrium exists in our model where both types of

firms develop code if (a) neither proposition 5.5.11 nor 5.5.12 are fulfilled, or (b) proposition 5.5.11 and 5.5.12 are fulfilled simultaneously. Together with the non-negativity conditions this leads to the conditions of the above proposition.

This section has analyzed code production equilibria for restricted licenses. For pure OS industries ($r = 0$) we have seen that firms develop OS code unless there is enough community code. For $x_{nc}^{os} = 0$ firms always produce OS code, see (5.13). Also in mixed OS/CS industries with $x_{nc}^{os} = 0$ commercial OS can exist.

5.6 Mixed Industries

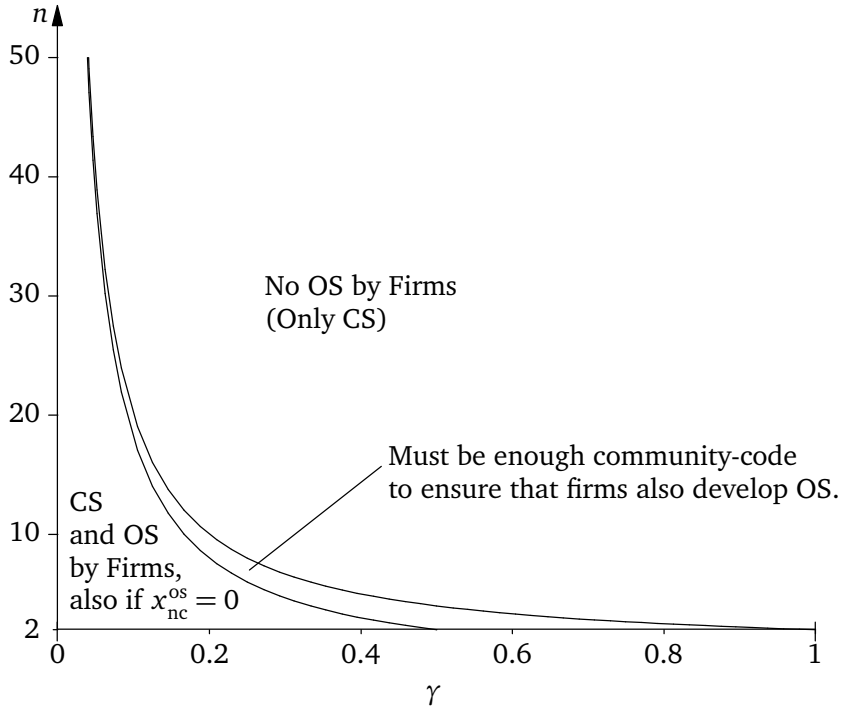
This section analyzes mixed industries, i.e. industries where firms develop OS and CS. Of course, we have to distinguish whether the OS license is liberal or restricted:

5.6.1 Liberal Licenses: OS and CS Development

With liberal licenses all firms can develop OS and CS code. Recall, however, that quality competition (θ) has a strongly negative impact on commercial OS production. This means that firms develop only CS code for all $\theta > 1/(n-1)$ (proposition 5.5.7). Absent community-code ($x_{nc}^{os} = 0$) moreover, firms do not develop OS code for $\theta > 1/(n+1)$ (proposition 5.5.8). This implies that mixed industries can only exist in the range $1/(n+1) < \theta < 1/(n-1)$ for appropriate non-zero values of x_{nc}^{os}. Mixed industries can however, exist for $\theta < 1/(n+1)$ even without non-commercial OS development.

Since $\theta = \gamma/(2-\gamma)$, the conditions $\theta < 1/(n+1)$ and $\theta < 1/(n-1)$ are equivalent to $\gamma < 2/(n+2)$ and $\gamma < 2/n$. Figure 5.4 provides a graphical impression of these two conditions. Except for very very concentrated (small n) industries, markets must be very separated (high horizontal product differentiation) for commercial OS to occur. For example, in the case of a $n = 10$ industry the two conditions for the horizontal degree of substitution are $\gamma < 0.1\bar{6}$ (always commercial OS) and $\gamma < 0.2$ (only enough x_{nc}^{os} ensures that firms develop OS). In the case of a $n = 100$ industry the two conditions

Figure 5.4: Liberal Licenses: OS and CS Code Development by Firms



are $\gamma < 0.0196$ and $\gamma < 0.02$ respectively. Finally, $\lim_{n\to\infty} 2/(n+2) = 0$ and $\lim_{n\to\infty} 2/(n) = 0$.

5.6.2 Restricted Licenses: Equilibrium of OS and CS Firms

This subsection analyzes the ratio of OS to CS firms, assumed free entry and exit. In the interest of generality, we do not postulate a specific entry process. For this we also ignore possible historical events that could give rise to lock in.[19] We analyze the condition for a stable mixed industry that resists further entry by OS and CS firms. A n-firm industry with z OS firms and r CS firms is an equilibrium if the incumbents earn profits $\pi_i \geq 0$ and

[19] The question of possible lock-ins, strategic OS-versus-CS decision of incumbents etc. is analyzed in the next chapter.

additional OS or CS entrant would earn negative profits. This condition is met where $\pi_i = 0 \, \forall i \in N$. This condition is both sufficient and necessary for large n.[20] We therefor define stability in terms of the the zero-profit condition[21]

$$\pi_{i\in Z} = p_{i\in Z} \cdot q_{i\in Z} - c_{i\in Z} - C = \pi_{i\in R} = p_{i\in R} \cdot q_{i\in R} - c_{i\in R} - C = 0. \quad (5.16)$$

We also concentrate on Nash-equilibria where both types of firm develop code: ($x^{cs*} > 0$, $x^{os*} > 0$). The reason is the following: If there is no community-code ($x^{os}_{nc} = 0$), then equilibria with ($x^{cs*} > 0$, $x^{os*} = 0$) and ($x^{cs*} = 0$, $x^{os*} > 0$) violate the zero-profit condition. The following lemmas 5.6.1 and 5.6.2 confirm these statements (we normalize β to $\beta = 1$ without loss of generality).

Lemma 5.6.1. *Given a Nash-equilibrium where only the CS firms develop software ($x^{cs*} > 0$, $x^{os*} = 0$), CS firms earn higher profits than OS firms provided that $\beta = 1$ and $x^{os}_{nc} = 0$.*

Proof by contradiction. Profits for $x^{cs} > 0$, $x^{os} = 0$ are $\pi_{i\in Z} = {(1-r\theta x^{cs})^2}/{h^2}$ and $\pi_{i\in R} = {(1+x^{cs}(1+z\theta))^2}/{h^2} - {x^{cs2}\phi}/{2}$ for $\beta = 1$ and $x^{os}_{nc} = 0$. There are two necessary conditions for $\pi_{i\in Z} > \pi_{i\in R}$:

(a) OSS-firms achieve positive prices if and only if $x^{cs} < {1}/{r\theta}$.

(b) $\pi_{i\in Z} > \pi_{i\in R}$ implies that $x^{cs} > {4(1+n\theta)}/{\left[\phi h^2 + 2\left(r^2\theta^2 - (1+z\theta)^2\right)\right]}$.

Conditions (a) and (b) are simultaneously fulfilled if and only if $\phi > {2(1+n\theta)^2}/{h^2}$. Inserting $\beta = 1$ and $x^{os}_{nc} = 0$ into proposition 5.5.11 leads to the condition $\phi < {2(1+n\theta)(1+(n-1)\theta)}/{h^2}$. But ${2(1+n\theta)^2}/{h^2} > {2(1+n\theta)(1+(n-1)\theta)}/{h^2}$. This shows that (a) and (b) can only be simultaneously fulfilled, i.e. that $\pi_{i\in Z} > \pi_{i\in R}$, if and only if a Nash-equilibrium with $x^{cs*} > 0$, and $x^{os*} = 0$ does *not* exist. □

[20] For small n the sufficient and necessary condition is: $\pi_i \geq 0 \, \forall i \in N$, $\pi_e < 0$ with $e \notin N$ is either a CS or an OS entrant. For $n \mapsto \infty$ the sufficient and necessary conditions converge to $\pi_i = 0$.

[21] A discussion of $\pi_i > 0 \, \forall i \in N$, $\pi_e < 0$ with $e \notin N$ is either a CS or an OS entrant can be found in Chapter 6.

Lemma 5.6.2. *Given a Nash-equilibrium where only OS firms develop software* ($x^{cs*} = 0$, $x^{os*} > 0$)*, OS firms earn higher profits than CS firms provided that* $\beta = 1$ *and* $x_{nc}^{os} = 0$.

Proof by contradiction. Profits for $x^{\text{cs}} = 0$, $x^{\text{os}} > 0$ are given by $\pi_{i\in Z} = {}^{(1+(1+r\theta)zx^{\text{os}})^2}/_{h^2} - {}^1/_2\phi z x^{\text{os}2}$ and $\pi_{i\in R} = {}^{(1-z^2\theta x^{\text{os}})^2}/_{h^2}$ for $\beta = 1$ and $x_{\text{nc}}^{\text{os}} = 0$. There are two necessary conditions for $\pi_{i\in Z} < \pi_{i\in R}$:

(a) CSS firms charge positive prices. This is true if and only if the condition $x^{\text{os}} < {}^1/_{(z^2\theta)}$ is met.

(b) $\pi_{i\in Z} < \pi_{i\in R}$ implies that $x^{\text{os}} > {}^{4(1+n\theta)}/_{[\phi h^2 + 2z(z^2\theta^2 - (1+r\theta)^2)]}$.

Conditions (a) and (b) are simultaneously fulfilled if and only if $\phi > {}^{2z(1+n\theta)^2}/_{h^2}$. Inserting $\beta = 1$ and $x_{\text{nc}}^{\text{os}} = 0$ into proposition 5.5.12 yields the condition $\phi < {}^{2z}/_{1+z} \cdot {}^{2(1+n\theta)(1+r\theta)}/_{h^2}$. But at the same time ${}^{2z(1+n\theta)^2}/_{h^2} > {}^{2z}/_{(1+z)} \cdot {}^{2(1+n\theta)(1+r\theta)}/_{h^2}$. We therefore conclude that (a) and (b) are simultaneously fulfilled, i.e. that $\pi_{i\in Z} < \pi_{i\in R}$, if and only if a Nash-equilibrium with $x^{\text{cs}*} = 0$, $x^{\text{os}*} > 0$ does *not* exist. □

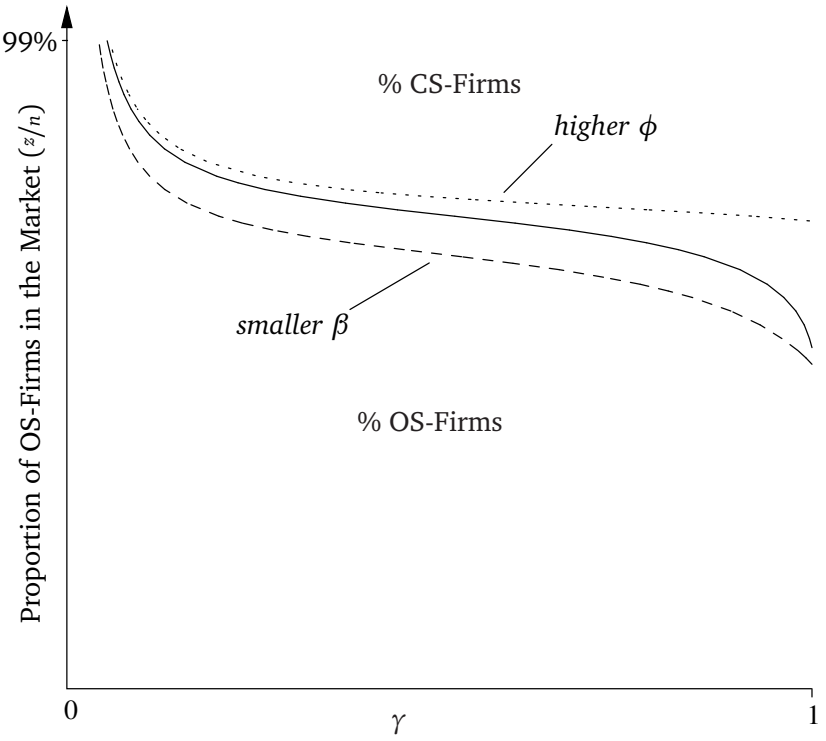
Lemma 5.6.2 also holds when community code is present. Lemma 5.6.2 has to be modified only slightly. Given sufficiently large amounts of community code, OS firms can survive in the market without contributing any code at all. Allegorically spoken, OS firms find themselves in "Cockaigne", the land of plenty, and get all code they need "for free". We will ignore this trivial case in what follows.

Our zero-profit condition (5.16) leads to

$$p_{i\in Z} \cdot q_{i\in Z} - c_{i\in Z} = p_{i\in R} \cdot q_{i\in R} - c_{i\in R}.$$

We use this condition to calculate mixed industries equilibria numerically. Figure 5.5 depicts the typical example of an industry with $n = 100$ firms. The solid line is the outcome for $\phi = 2$, $\beta = 1$, and $x_{\text{nc}}^{\text{os}} = 0$. ($x_{\text{nc}}^{\text{os}}$ was set equal zero to ensure that OS firms never find themselves in "Cockaigne". The impact of $x_{\text{nc}}^{\text{os}} > 0$ is explained below.) As the reader can confirm by inspection, the proportion of CS firms decreases when the products produced in stage two are close substitutes. Figure 5.6 shows the total market share of OS and CS products (bundles). For industries with with a low degree

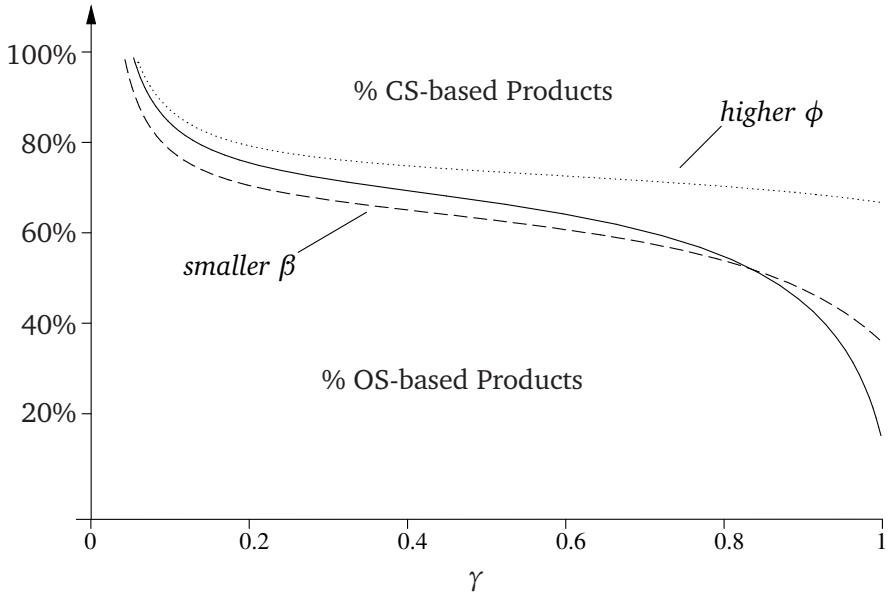
Figure 5.5: Proportion of OS Firms



of horizontal product differentiation (large γ), CS-based products have a bigger market share than OS-based products. For γ close to one, 80% or more of the products are based on CS, while less than half of the firms in the market are CS firms. In other words: a few 'big' CS firms compete with many 'small' OS firms. Furthermore, CS firms offer higher quality (more code per bundle). Changing the parameters ϕ and β changes the proportion of OS firms and their market share. However, CS firms still offer higher quality. Figures 5.5 and 5.6 depict the impact of differing ϕ and β on OS firm proportion and market share.

The dotted line represents the case of $\phi = 5$, i.e. when the marginal costs of software development increases more steeply. As result, the number of OS firms in the industry and their market share is higher. The reason for

Figure 5.6: Marketshare of OS- vs. CS-based Products



this is that high ϕ makes software development more costly as costs increase more steeply. This makes OS's cost-sharing and quality-cooperation benefits more attractive.

The dashed line represents a lower quality stage two product, e.g. mobile phones with less hardware features (we have set $\beta = 0.8$). Recall, that the quality of the bundles ($\alpha_i = \beta + x_i$) directly affects demand: $p_i = \alpha_i - q_i - \gamma \sum_{j \neq i} q_j$. Thus, a lower β means that the complementary private good is less important for generating revenues. This makes the software-decision more important. As one would expect, fewer firms choose OS business models in this scenario. On the other hand, the remaining OS firms have a stronger incentive to develop code because of quality competition with the CS firms. This explains why the market share of OS-based products is higher for large γ than in the $\beta = 1$ case. Here the OS firms produce more code (offer higher quality) than they would do in the $\beta = 1$ case.

Finally the impact of community-code can be analyzed with numerical examples. The effect is straightforward: $x_{\text{nc}}^{\text{os}} > 0$ makes OS business models

more attractive. This increases the number of OS firms in the industry along with their total and individual market share. Furthermore, firms stop developing OS code even for small x_{nc}^{os} if the products becomes close substitutes. In these cases OS firms survive in the market without own code development. Here, OS firms do not contribute to the OS software but just use it.

5.7 Summary and Outlook

The chapter uses a general model to analyze the *economics of open versus closed source business models*. In stage one firms develop software, as OS or CS code, or as a mix of OS and CS code. In stage two firms bundle this 'stage one'-software with complementary products (hardware, service, or proprietary software) and compete. Competition in stage two is modeled as oligopolistic competition. We allow for *horizontal* product differentiation. Furthermore, firm i's software developed in stage one affects quality and hence increase consumers' willingness to pay for q_i. This yields *vertical* product differentiation. Using this framework we have shown:

- General two-stage models that combine aspects of non-cooperative R&D with the theory of differentiated oligopolies provide a convincing explanation of why firms adopt OS business models. Specifically, OS enables firms to cooperate on quality and therefore avoid quality competition.
- If licenses allow a direct mix of OS with CS code, firms only develop OS where the degree of quality competition is low. Otherwise a public good dilemma occurs: firms use OS code if exists but only produce CS code. On the other hand, restricted licenses ensure commercial OS output.
- Each firm's CS output is always a strategic substitute for every other firms' CS code. However, firm's OS output can be a strategic complement to other firms' OS code.
- Assuming free entry and restricted OS licenses, mixed industries (OS and CS firms) equilibria exist. Here OS firms offer lower quality than

their CS rivals. Where the products are close substitutes, CS-based products have the larger market share.

The chapter does not calculate welfare. However, the fact that the OS firms are the low-quality providers in industry equilibrium provides a first hint that there are too many OS firms in the market compared to social optimum. The chapter is also not explicit about how entry occurs and thus ignores possible historical events like lock-ins. Finally, incumbents' OS versus CS strategic choices are not analyzed. It is therefore possible for industries to become stuck in pure rather than mixed equilibria. These issues are analyzed in Chapter 6.

Finally, the chapter only consider firms' simple economic incentives. Social interactions and community norms are not taken into account. It is well known, that the OS community pays attention to what firms do and expect at least some kind of contribution from the firms. Thus, such community norms can ensure that firms that fail to contribute will be sanctioned as community members stop cooperating or migrate to other projects (see also Section 4.3.3.2). This suggests that firms may contribute OS code more often than the model predicts. However, it still holds that our simple Cournot logic provides a complete explanation of why firms contribute to OS. this is particularly true for OS business models in industries where there is no non-commercial community. Our model also underscores the importance of formal institutions, here: the type of OS license.

6 The New (Commercial) Open Source: Does It Really Improve Social Welfare?

6.1 Introduction*

The last chapter has analyzed the strategic nature of OS vs. CS business models, including the role of OS licenses. In the present chapter we focus on welfare and discuss possible government interventions. Today, many governments support OSS and are in favor of the OS principle in general. The OS principle seems to be an alternative to the traditional incentive schemes like copyright-based CS, patents, prizes, grants, contract research, etc. The emergence of OS methods for producing software in the 1990s surprised and delighted observers. OS seemed to avoid CS software's worse feature—charging consumers a royalty for information that could theoretically be distributed at zero cost. This made it natural to ask whether OS could drastically improve welfare compared to CS. At first, this was only an intuition. Early explanations of OS were either ad hoc ("altruism") or downright mysterious (e.g. a post-modern "gift economy," see Raymond, 1998). Absent a clear model of OS, no one could really be certain how much software the new incentive could deliver, let alone whether social welfare would best be served by OS, CS, or some mix of the two.

The past decade has seen considerable progress. Following Lerner and Tirole (2002)'s seminal article, economists showed that real world OS collaborations rely on many different incentives such as education, signaling, and reputation (see Section 1.3.1). Furthermore, they constructed detailed mathematical models of each mechanism (Maurer and Scotchmer, 2004). The problem for policymakers was that the new models—despite a family

*This Chapter is based on von Engelhardt and Maurer (2010)

resemblance—were all different and sometimes yielded contradictory insights. Worse, their predictive power was limited. Saying that certain OS projects were driven by a desire for reputation was one thing. Saying how much desire actually existed, let alone how much software it would generate, was another. Furthermore, the existence of multiple, competing models was disabling for policymakers. Government interventions that made sense for one set of open source incentives were likely to be irrelevant or even counterproductive for others.

The situation today is much improved. The reason is that the OS phenomenon has itself become more uniform. More and more firms have begun to engage in OSS development. As already mentioned in the last chapter's introduction, much of the last years' growth in OSS projects is commercially motivated. The number of companies in these OS communities typically ranges from a few dozen to many thousands. The following examples provide some idea of this range: The Lamp Stack software suite (see Section 1.2.3) is an example for small communities. Here, development is supported by a relatively small number of corporations. In 2001, IBM opened its Eclipse development tool and created an independent foundation to manage further OS development in 2003. Eclipse is an example for mid-size communities as 115 companies had joined the foundation as of 2006. Thousands of programmers developed the Linux code base so that they could develop custom software solutions for clients. This is an example for large communities. Many firms, ranging from large cooperations to small and medium sized enterprises, use Linux as basis for their business model (e.g. distributors, embedded software, etc., see Section 1.2.3). Therefore this chapter makes use of the last chapter's model of how firms decide how much OSS and CSS to produce. We analyze welfare implications of these decisions, and possible government options for improving these outcomes.

Like earlier contributions, we find that the amount of OS produced reflects a balance between OS firms' ability to share costs and CS firms' greater ability to appropriate benefits to consumers. Depending on this balance, Pure-OS industries may or may not be welfare-superior to Pure-CS industries while Mixed-OS/CS industries are almost always welfare-superior to both. Unlike earlier contributions, our general model lets us systematically explore these competing effects for arbitrary combinations

of substitutability between competing proprietary products, and numbers of competing OS and CS firms.

Crucially, we find that OS firms hardly ever realize the full welfare benefits of cost sharing. The reason is that OS firms (unlike CS firms) share all software so that no firm can offer better quality than any other firm. This leads to a quality cartel effect not seen in earlier models that drastically suppresses of OS code production. Paradoxically, then, we find that OS firms deliver relatively modest welfare gains unless and until CS firms are present in sufficient numbers to enforce quality competition. We also use our model to calculate which OS:CS firm ratios are stable against entry. We find that (a) many Pure-OS and Pure-CS markets are stable so that welfare-improving mixed OS/CS markets never arise, and (b) stable OS/CS mixed markets hardly ever contain enough CS firms to enforce optimal quality competition.

Understanding these effects matters. Most national governments already purchase large amounts of OS software and this spending is often explicitly justified as an attempt to promote OS over traditional CS code production models. We show below that such interventions can actually decrease social welfare. More generally, we use our model to evaluate welfare effects for a wide spectrum of possible interventions including taxation, direct funding of OS-development, and government procurement preferences.

The balance of this chapter proceeds as follows. Section 6.2 describes how various governments support OS development and set incentives to promote OS companies. Section 6.3.1 analyses the profit-maximizing output for arbitrary numbers of firms operating in Pure-CS, Pure-OS, and mixed OS/CS industries, and then shows how specific output decisions translate—through costs incurred and quality delivered to consumers—into net welfare. Section 6.4 identifies the conditions under which industries endowed with an initial mix of OS and CS firms are stable against entry. Crucially, it finds that few, if any of these equilibria are welfare-optimal. Section 6.5 examines several strategies that government could potentially use to intervene in these markets. Section 6.6 discusses the generality of these results and the extent to which more complicated models could lead to different conclusions. Finally, Section 6.7 presents a brief conclusion.

6.2 Background: Government Interventions

Governments are plainly intrigued with OS and have repeatedly flirted with various schemes to promote it. Probably the most comprehensive survey of existing and proposed initiatives is found in CSIS (2008). It reports that governments have experimented with a wide variety of incentives to promote OS companies. These include:

Procurement Preferences. Governments purchase large amounts of software and can potentially use these purchases to promote OS over CS and vice versa. At least sixteen countries have considered mandatory polices that would require government agencies and/or state-owned companies to purchase OS solutions whenever possible.[1] Softer versions of these proposals speak of "preferences" for OS when its performance is comparable to CS. To date, at least ten national governments have adopted some version of these proposals[2] along with many state and local governments. High government adoption rates of OS in still other countries (e.g. France) suggests that unofficial preferences also exist.

Tax Incentives. Singapore offers tax breaks to firms that use LINUX operating systems.

Government Funding. Hong Kong offers funding for companies that adopt or use OS. Israel offers grants of up to $100,000 to start-up companies that use and develop OS. Governments have also funded a variety of institutes, projects, and private-public collaborations to develop[3] and facilitate user adoption[4] of OS software.

[1] Argentina, Belgium, Brazil, Bulgaria, Chile, Colombia, Israel, Italy, Netherlands, Ukraine, Finland, Portugal, Peru, and Venezuela.

[2] Australia, Belgium, Brazil, China, Malyasia, the Netherlands, Peru, South Africa, Spain, and Venezuela,. Conversely, the UK, Canada, Germany and Slovenia have said that they will choose between OS and CS solely on the technical merits.

[3] China, Finland, Japan, South Korea, France, India, Slovakia, Spain, Thailand, Venezuela, Vietnam

[4] China, Czechoslovakia, Israel, South Africa, Taiwan, Cambodia, South Korea, Japan, Netherlands, Philippines, Thailand, Vietnam.

Grants Policy. Government research grants to academia and industry frequently require dissemination plans when software is produced. OS is by far the easiest way to meet these obligations. More formally, the United Kingdom has adopted a "default position" that government-funded software should be released under OS licenses (CSIS, 2008).

While scholars have occasionally explored the case for such interventions (e.g. Schmidt and Schnitzer, 2003), their welfare analyses have generally focused on the impact of government policy on OS collaborations driven by altruism, reputation, signaling, and other traditional incentives. Not surprisingly, these studies usually assumed that government spending could do little to influence OS code production. This assumption clearly needs to be revisited in an era when commercial incentives dominate OS production. Similarly, government policy in recent years has increasingly evolved from simple OS-promotion schemes to "a search for business models that can profitably blend open and proprietary processes and products." (CSIS, 2008) However, earlier articles say relatively little about how the two sectors interact or how these interactions can be managed to improve welfare. We fill this gap by exploring how various government interventions including taxation, funding, and purchasing preferences influence output (and indirectly, welfare) for a very general commercial models in which OS and CS firms interact with one another.

6.3 A Simple Commercial OS Model: Calculating Output and Welfare

The analysis makes use of the model developed in the last chapter. Therefore we will refer to the reaction functions etc. presented in Chapter 5. However, in the current chapter we focus on the case of restricted OS licenses. The reason is that if OS licenses are liberal, commercial OS development exists only for a small set of parameter constellations (low n and/or low γ), see Figure 5.4 in Section 5.6.1. Furthermore, the sake of simplicity we have normalized β to one.[5] Finally, as we concentrate on

[5] Values different than $\beta = 1$ do not change the results qualitatively. However, for $\beta > 1$ OS becomes more attractive.

commercial OS, we abstract from a non-commercial OS community. Thus, like in Section 5.6.2, we set $x_{\text{nc}}^{\text{os}} = 0$.

While Chapter 5 focus on the strategic interactions, we will now deepen the analysis of OS and CS code output. This is the basis of understanding the welfare effects in Section 6.3.2. For now, we treat the number of OS and CS firms as a free parameter without asking whether they represent equilibrium outcomes in real markets. (We will return to this question in Section 6.4)

6.3.1 How Much OS and CS Software Does the Market Supply?

6.3.1.1 A Pure-CS Industry

Consider first an industry in which no OS firms exist. How much CS is produced? In general, the answer depends on firms' strategic interactions, i.e. on how Firm A reacts to Firm B's decision to produce code. In a Pure-CS industry the reaction function 5.10 simplifies to

$$R_{i\in R}^{\text{cs}} = \frac{(1+(n-1)\,\theta)\left(1-\theta\sum_{j\neq i} x_j^{\text{cs}}\right)}{\frac{1}{2}\phi h^2 - (1+(n-1)\,\theta)^2}. \tag{6.1}$$

Recall that here firms' software development decisions are strategic substitutes. The size of this effect depends on θ, i.e. the extent to which firms compete on quality. Furthermore, an industry composed of n identical firms obeying (6.1) has the following Nash equilibrium:[6]

$$x^{\text{cs}*} = \frac{(1+(n-1)\,\theta)}{\frac{1}{2}h^2\phi - 1 - (n-1)\,\theta} \tag{6.2}$$

As before—see Section 5.5.4—CS code development is suppressed by intense quantity competition, i.e. the presence of h in the denominator. Conversely, quality competition—$(n-1)\theta$—increases equilibrium code-output by making the numerator larger and denominator smaller.

[6] Compare this to (5.12), but note that in (5.12) is $\psi = \frac{1}{2}h^2\phi - (1+(n-1)\,\theta)(1+z\theta)$.

As this is useful for understanding the welfare discussion later on, we now discuss the impact of γ and n. Both h and θ depend on the substitutability (γ) of the 'stage two'-products, while h also depends on the number of competitors (n). The net effect of an increase of n is straightforward: it decreases equilibrium code-output. However, changes in γ have a positive impact on both quantity competition and quality competition. Because quality and quantity competition exert opposing effects on CS output the net effect is more complicated. Figure 6.1 plots closed source (x^{cs}) production as a function of γ.

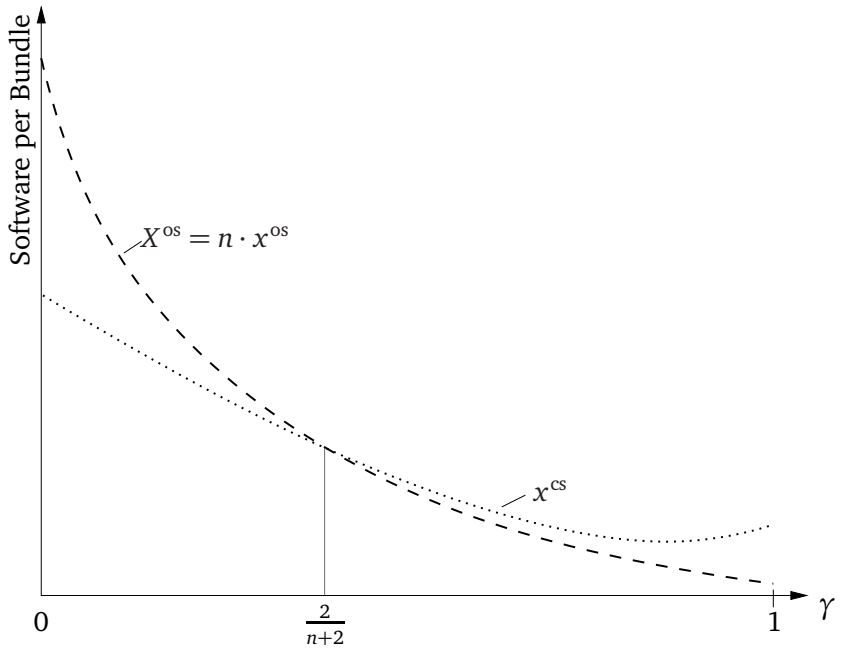
For low-to-moderate values of γ the amount of CS code produced is mainly determined by quantity competition (h), i.e. firms' ability to extract extra profits when quality increases. This ability is highest when products have no substitutes ($\gamma = 0$ yields $h = h^{\min} = 2$) so that each firm can set monopoly prices unconstrained by competition. It steadily erodes as substitutability (γ)—and hence h—increases.

There is also a second effect determined by θ. For very large γ products are nearly identical so that even small quality differences can lead to large swings demand for or against a particular bundle. This makes quality competition extremely important. Specifically, CS firms find themselves in a kind of Arms Race or Prisoner's Dilemma in which each firm invests in quality to prevent every other firm from taking its business. This drives industry profits toward zero. This effect becomes so large for γ larger than ~ 0.9 that software production in a Pure-CS industry actually starts to rise again.[7]

Result 1. Software output in Pure-CS industries is suppressed by quantity competition but boosted by quality competition. Specifically, output fall as the number of competing firms (n) and/or product substitutability (γ) increases, hence h increases. However, at high γ quality competition (high θ) between firms actually increase output by creating an Arms Race dynamic in which each firm invests in quality to keep rivals from taking its business.

[7] Firms do not, of course, invest to the point where they would earn negative profits. For this reason, the effect is stronger in concentrated (small n) industries where firms possess sufficient market power to extract oligopoly profits. More precisely, CS starts rising at $\gamma = \left(n-5\sqrt{9+(n-2)n}\right)/2(n-2)$. This point is located at $\gamma = 0.8956$ for $n = 4$ and is higher for all other n. For large n software production only starts to rise near $n = 1$.

Figure 6.1: Software per Bundle in Case of Pure-OS and Pure-CS (stylized illustration)



6.3.1.2 A Pure-OS Industry

Now consider the opposite case where no firms produce CS. Compared to CS industries, OS introduces two new effects. First, firms share cost. This means that the average per-firm development costs are lower for OS compared to CS firms. Second, OS firms share all software. This means that a firm's decision to invest in OS software not only makes its own bundles more attractive but also—contrary to the CS case—strengthens its competitors. Furthermore, the existence of shared software guarantees that no OS firm can offer better quality than any other OS firm. This suppression of quality competition implies that firms in Pure-OS industries always earn higher profits than firms in Pure-CS industries for a given number of incumbents. The same logic also implies that more firms will enter Pure-OS than

Pure-CS industries. Ceteris paribus we therefore expect Pure-OS industries to have more incumbents when free entry is present.

For a Pure-OS industry the reaction function 5.11 simplifies to

$$R^{os}_{i\in Z} = \frac{1 - \left(\frac{1}{4}\phi h^2 - 1\right)\sum_{j\neq i} x^{os}_j}{\frac{1}{2}\phi h^2 - 1}. \tag{6.3}$$

Crucially—and unlike the Pure-CS case, see (6.1)—Firm B's decision to produce software no longer depends on quality competition (θ) from other firms.[8] It does, however, depend on cost-sharing. This leads to several important differences from the CS case.

- *Strategic Complements vs. Substitutes:* For CS Firms, increased software output by Firm B always suppresses software for Firm A. By contrast, as mentioned in Section 5.5.2, the result for OS firms is ambiguous. OS implies code- and cost-sharing. The net result is that firm decisions sometimes become strategic complements, i.e. that Firm B's decision to develop more software can cause Firm A to increase its code output and vice versa. This occurs when the marginal cost of software production increases slowly (low ϕ) or competition is modest (low h). More formally, OS investments become strategic complements in our model when $h^2\phi < 4$, see proposition 5.5.2.

- *Quality Cartel:* We have seen that quality competition among CS firms leads to Arms Races at high γ in which firms continue to invest in software until rising marginal costs wipe out any profits that could have been earned from increased demand. By contrast, OS firms do not compete on quality. This means that they face no Arms Race, so that code output is suppressed to levels slightly below those that would be expected under a formal "quality cartel" charged with setting output to whatever level maximizes total industry profit.[9]

[8] See also the short note on this in Section 5.5.4.

[9] The difference stems from the fact that individual OS firms cannot recover the positive impact that their software investment confers on the profits of every other OS firm. A formal quality cartel would internalize this externality. For more details see the Appendix D.1

As before, output decisions by n identical firms lead to a Nash equilibrium in which each firm produces the following amount of software:[10]

$$x^{os*} = \frac{1}{\frac{1}{4}\phi h^2 (1+n) - n} \tag{6.4}$$

This implies that total industry-wide output ($X^{os} = nx^{os}$) is

$$X^{os*} = \frac{n}{\frac{1}{4}\phi h^2 (1+n) - n} \tag{6.5}$$

except in those cases where OS development would exceed the cutoff, i.e. deliver more code than society can use.[11] This leads to a third fundamental difference between the Pure-OS and Pure-CS models:

- *Quantity Competition and Cost-Sharing.* As in the Pure-CS case, the amount of software produced in Pure-OS industries depends negatively on quantity competition and is greatest at low γ, see Figure 6.1. Now, however, there is a second effect. Because of shared development costs, Pure-OS industries are able to offer more software per bundle than Pure-CS industries as long as $\gamma < 2/(n+2)$.

- *Quantity vs. Quality Competition.* We have seen that quality competition gradually replaces quantity competition as the most important factor in determining CS output at high γ. Pure-OS industries, however, are able to suppress quality competition through code-sharing (quality cartel). This explains why Pure-CS industries deliver more software than Pure-OS industries in our model for $\gamma > 2/(n+2)$ (see Figure 6.1) or, equivalently, $h > 4 - 3\gamma$.

The balance between quantity competition and cost-sharing is superficially consistent with Llanes and de Elejalde (2009) and Henkel (2006)'s findings that OS business models are most profitable where quantity competition is low so that cost-sharing dominates. However, these earlier analyses are incomplete to the extent that they fail to consider the suppression

[10] Compare this to (5.13), but note that in (5.13) is $\chi = \frac{1}{4}\phi h^2 (1+z) - z(1+r\theta)^2$.

[11] As before in Chapter 5 we focus on cases where OS development does *not* exceed the cutoff.

of quality competition in Pure-OS industries. This latter effect systematically reduces the amount of OS output that would otherwise be expected from a simple balance of appropriability and cost-sharing.

Result 2. Cost-sharing allows Pure-OS industries to produce more software than Pure-CS industries so long as quantity competition is modest. However, production falls steeply as greater product substitutability (γ) leads to increased quantity competition between firms. Unlike CS firms, OS firms do not compete on quality (the 'quality cartel effect'). For this reason a Pure-OS industries offer less software per bundle than Pure-CS industries for $\gamma > 2/(n+2)$.

6.3.1.3 A Mixed OS/CS Industry

Finally, consider the case where both OS and CS firms exist. As before, CS firms still react to CS firms and OS firms still react to OS firms. With $\beta = 1$ and $x_{\text{nc}}^{\text{os}} = 0$ (5.12) and (5.13) become[12]

$$x^{\text{cs}} = \frac{(1+(n-1)\theta)\left(1-z^2\theta x^{\text{os}}\right)}{\psi} \tag{6.6}$$

$$x^{\text{os}} = \frac{(1+r\theta)(1-\theta r x^{\text{cs}})}{\chi} \tag{6.7}$$

where z is the number of OS-firms and r is the number of CS-firms so that $n = r + z$. Like in Chapter 5, for convenience we replace the denominator of x^{cs} and x^{os} with ψ and χ.[13] Now, however, CS firms also react to OS firms and vice versa. The overall Nash-equilibrium is thus a simultaneous solution of these two functions.

As before, CS firms react to increased software development by other firms as a strategic substitute. Since OS firms compete with CS firms on quality, they also see increases in CS production as strategic substitutes. This is represented by $-\theta r x^{\text{cs}}$ in the numerator of (6.7). The situation regarding the strategic interaction among OS firms is different. The presence

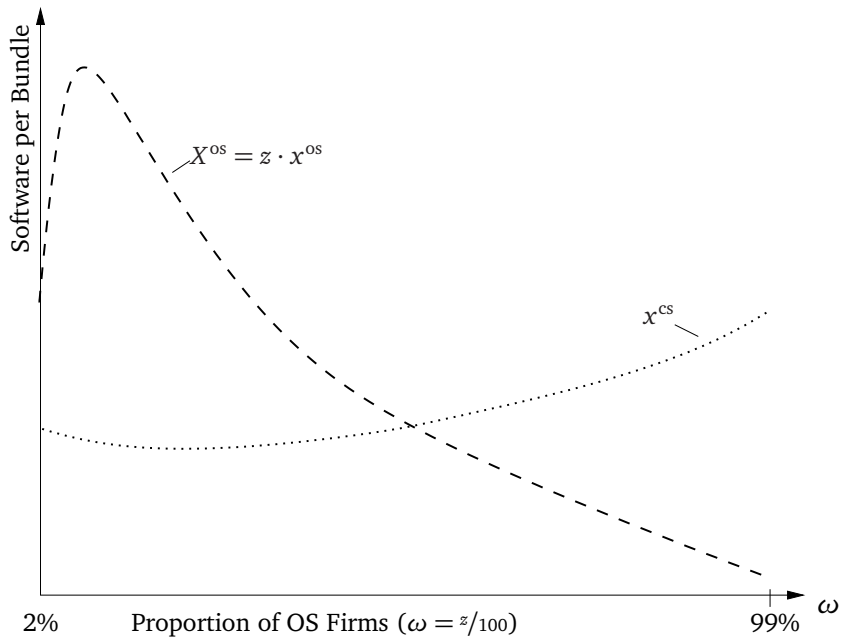
[12] As before, we exclude cases where OS would exceed the cutoff, i.e. deliver more code than society can use. Formally, we restrcit our analysis to $\phi > 4 \cdot z/(z+1) \cdot (1+r\theta)^2/h^2$.

[13] $\psi = \frac{1}{2}h^2\phi - (1+(n-1)\theta)(1+z\theta)$ and $\chi = \frac{1}{4}\phi h^2(1+z) - z(1+r\theta)^2$.

of CS firms prevents OS firms from cartelizing around a low level of quality. Instead, OS firms must compete on quality against these CS outsiders to the cartel. This makes the ability to share costs more valuable to OS firms. Ceteris paribus, this increases OS output and increases the strategic complements effect among OS firms. This explains why OS-investments are strategic complements for $h^2\phi < 4$ in Pure-OS industries but $h^2\phi < 4(1+r\theta)^2$ for mixed OS/CS industries (see also proposition 5.5.2).

Because of these interactions, the detailed behavior of a mixed OS/CS industry depends on the ratio of OS to CS firms. Figure 6.2 parameterizes this as the proportion of OS firms $\omega = z/n$ and shows how much software consumers receive per bundle in the typical example where $\gamma = 0.5$ and $n = 100$. Where OS firms are a small minority ($\omega \sim$ a few percent), they

Figure 6.2: Software per Bundle in a Mixed Market ($n = 100, \gamma = 0.5$)



face strong quality competition from CS firms. This encourages them to use their cost-sharing advantage to produce large amounts of OS software. Increased opportunities for cost sharing continue to dominate diminished

quality competition until the proportion of OS firms reaches $\omega \sim 0.08$. Thereafter, however, the declining number of CS firms suppresses quality competition so that OS code production falls. For large ω the OS quality cartel is so strong that OS firms produce very little code.

Because CS firms do not share costs, they cannot possibly match the maximum potential software output that OS firms can achieve. Furthermore, as already mentioned, CS firms react to OS development as a strategic substitute. As long as OS production is high, therefore, CS firms will specialize in selling low-quality bundles at a low price. The situation is reversed as OS productions declines. As a result, CS firms replace OS firms as the industry's high-quality, high-priced providers above $\omega \sim 50\,\%$.

Result 3. Quality competition from CS firms in mixed OS/CS industries mitigates the quality cartel effect that suppresses production in Pure-OS industries. As a result, OS-software production for suitably chosen OS:CS ratios is dramatically higher than that found in otherwise comparable Pure-CS or Pure-OS industries. If the OS:CS ratio is low, OS firms are the industry's high-quality, high-priced providers. This situation is reversed when the industry hosts many OS firms and only a few CS firms.

6.3.2 Welfare Implications

We now know how much software a Pure-OS, Pure-CS, or Mixed-OS/CS industry produces. This allows us to calculate firm profits and consumer utility, which in turn enables us to analyze welfare. More specifically, producer surplus is given by total industry profits and consumer surplus for differentiated oligopolies is given by $1/2\left[(1-\gamma)\sum q_i + \gamma(\sum q_i)^2\right]$ (see Hsu & Wang 2005). This yields the following general welfare function:

$$W = \frac{1}{2}\left[(1-\gamma)\sum q_i + \gamma\left(\sum q_i\right)^2\right] + \sum \pi_i \tag{6.8}$$

6.3.2.1 Pure-OS vs. Pure-CS

We begin by comparing welfare under a Pure-OS regime against a Pure-CS case. For convenience, we focus on the difference in welfare between a Pure-OS and a Pure-CS world. Prices and quantities of Pure-OS and

Pure-CS are given by $q = p = {}^{(1+nx^{os})}/_{h}$ and $q = p = {}^{(1+x^{cs})}/_{h}$ respectively. The *difference* in welfare between a Pure-OS and a Pure-CS world ($\mathcal{W}_n = W_n^{os} - W_n^{cs}$) is given by

$$\mathcal{W}_n = \underbrace{\frac{n}{2}(1+h)\frac{(1+nx^{os})^2-(1+x^{cs})^2}{h^2}}_{\textit{quality difference}} - \underbrace{\frac{n}{2}\phi\left(nx^{os2}-x^{cs2}\right)}_{\textit{cost difference}}. \quad (6.9)$$

Note that the welfare difference consists of two components. The first reflects the *quality difference* and the second term represents the *cost difference* between a Pure-OS and a Pure-CS world. (For more details on the welfare functions see Appendix D.2)

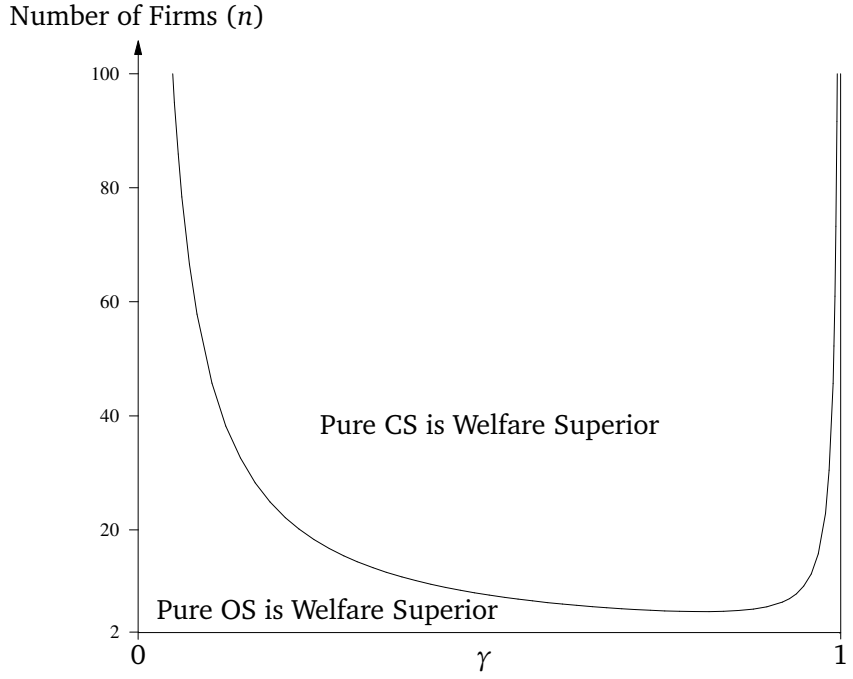
We start by examining welfare at $\gamma = {}^{2}/_{(2+n)}$ where, as we have already seen, Pure-OS and Pure-CS systems produce the same amounts of code (see Figure 6.1) so that the quality difference between firms' 'stage two'-products is zero. Here, the remaining cost difference term makes OS welfare superior. This is because OS firms can share code whereas each CS firm must create its own code base *de novo*. This wasteful duplication of effort is variously described as "business stealing" or "me-too products" in the literature (Henkel and von Hippel, 2005).

OS's welfare-superiority diminishes for small values of γ. This is because Pure-OS industries produce much more code (see Figure 6.1) and therefore incur higher costs. However, the effect is never large enough to overcome the quality and cost-sharing advantages associated with OS. For this reason, Pure-OS industries remain preferable to Pure-CS industries in our model for all γ less than ${}^{2}/_{(2+n)}$. Significantly, this statement does not depend on ϕ and therefore holds regardless of detailed assumptions (e.g. Brooks' Law) about how quickly the marginal cost of software production rises.

The situation is more ambiguous for γ larger than ${}^{2}/_{(2+n)}$. At first OS's welfare-superiority erodes with increasing γ because of the greater code production associated with Pure-CS industries (see Figure 6.1). For $\gamma > {}^{2}/_{(2+n)}$ Pure-CS delivers higher quality. For moderate large γ this effect dominates the cost advantages of shared OS production so that CS also delivers superior welfare. On the other hand, we have seen that code output in Pure-CS industries increases sharply for very large values of γ. This can produce such large cost increases that a Pure-OS industry is once again welfare-superior.

Figure 6.3 summarizes these results for a representative numerical example in which we have set $\phi = 2$.[14] (We solve $\mathcal{W}_n = 0$ for n, with $\phi = 2$ and $\mathcal{W}_n$, x^{cs} and X^{os} given by (6.9), (6.2) and (6.5) respectively. The result is plotted in Figure 6.3.) Note in particular that OS is welfare-dominant in highly concentrated industries (low values of n), for limited substitutability (low values of γ), and situations where both n and γ are moderate. More concisely, OS is welfare-superior where $h = 2 + (n-1)\gamma$ is small. Furthermore, OS is also welfare-dominant for very high values of γ. Pure OS are thus superior both for low quantity competition and very high quality competition.

Figure 6.3: Welfare Superiority of OS- vs. CS-only for $n = 2 \ldots 100$ ($\phi = 2$)



[14] Our results do not change significantly for values different than $\phi = 2$, see Appendix D.3

Result 4. We find (a) that Pure-OS industries are welfare superior to Pure-CS industries for low values of quantity competition, (b) that Pure-CS industries are welfare-superior to Pure-OS industries for high values of quantity competition, and (c) that Pure-CS industries again become welfare-superior to Pure-OS industries when competing products are very close substitutes so that quality competition is intense.

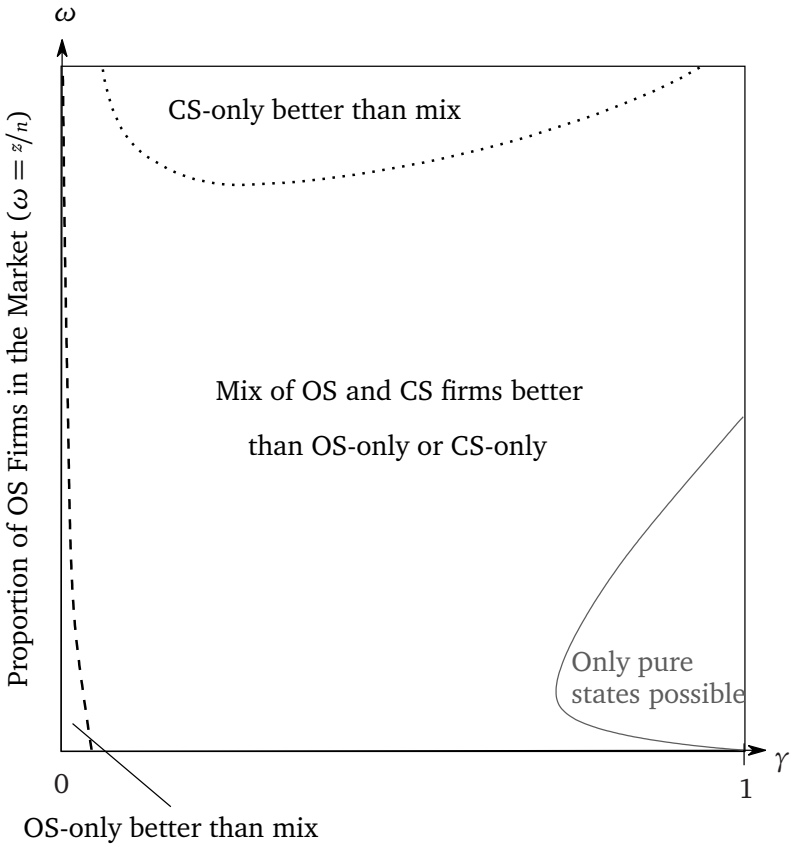
6.3.2.2 Mixed OS/CS-Industries

We now extend our welfare analysis to include arbitrary mixes of OS and CS firms where the proportion of firms creating OS is given by $\omega = z/n$. We want to know whether Mixed OS/CS, Pure-CS, or Pure-OS industries generate more welfare. We therefore use (6.8) to calculate welfare for each pair (ω, γ), and compare this against our results for Pure-OS and Pure-CS industries. (For more details on the welfare function see Appendix A.3). Figure 6.4 depicts our results for industries containing $n = 100$ firms, $\phi = 2$. (We plot the welfare of Pure-CS, Pure-OS and Mixed OS/CS with $\phi = 2$ and $n = 100$ depending on ω and γ. The topview of the resulting 3-D plots yields Figure 6.4.)

Figure 6.4 contains four distinct regions. The largest region consists of mixed states that are welfare-superior to the corresponding Pure-OS or Pure-CS state. Furthermore, readers can confirm by inspection that such welfare-superior mixed states exist for all values of γ. Thus, provided that ω (the proportion of OS-firms) can be suitably chosen mixed industries are the better choice. While we have depicted this situation for the special case of $n = 100$, this statement is in fact generally true for all large n, while for very concentrated industries (small n) the situation can differ (For some low value of γ welfare-superior mixed states might not exist as Pure-OS is superior for all ω. See Figure D.2 and Figure D.3 in Appendix D.4).

In practice, of course, policymakers may have no direct control over the proportion of OS-firms (ω) in the market. In this case, pure states will sometimes offer higher welfare than mixed ones. This is reflected in the two smallest regions of the graph which are, in effect, much-shrunken versions of the Pure-OS and Pure-CS states depicted in Figure 6.3. First, consider the high ω region where OS firms greatly outnumber CS firms. Here, Pure-CS states are welfare-superior to mixed states for the same rea-

Figure 6.4: Welfare-Comparison of Pure vs. Mixed Cases ($n = 100, \phi = 2$)



sons that they dominate Pure-OS states. This region shrinks and eventually disappears as industry becomes more concentrated so that OS firms can appropriate value despite high γ. (see Figure D.2 and Figure D.3 in Appendix D.4). Second, Pure-OS is welfare-superior to mixed states in low ω/low γ cases where CS firms greatly outnumber OS firms and products have low substitutability. Here, OS firms can recover their investments even without intellectual property protection. As a result, cost-sharing dominates the welfare analysis so that Pure-OS states become superior. This region grows for concentrated (low n) markets until OS firms can recoup their invest-

ments even for moderate γ. The result is a drastically simplified graph in which the welfare-dominant regions are either Mixed-OS/CS or Pure-OS states (Figure D.3, Appendix D.4).

Finally, consider the lower right-hand corner of Figure 6.4. Here, only pure states are possible because (a) strong competition limits OS firms' ability to appropriate profits from improved products, and (b) the small number of OS firms limits the savings otherwise available from shared development. This means that the profit-maximizing investment for OS-firms facing CS-competition is to produce no software at all.[15] As a result, the only meaningful choice is between Pure-CS and Pure-OS-states. Here, the Pure-CS state turns out to be welfare-superior (see Figure 6.3 for the case of $n = 100$).[16] This region also disappears for highly concentrated industries where competition is weak (Figure D.3, Appendix D.4).

Welfare also differs within the mixed regions. For example, welfare in an $n = 100$ industry with $\gamma = 0.5$ reaches its maximum when OS-firms account for 20 % of all firms: While producer surplus reaches its maximum at $\omega = 99\,\%$, consumer surplus has an inverse U-shape with its peak at $\omega = 18\,\%$. The shape of consumer surplus is driven by the quality of the bundles available in the market and is maximized at a reasonable mix of high a few quality high-priced OS-bundles and many low-quality, low-priced CS-bundles. Thus, total welfare is maximized at a point where the OS firms select the much higher output levels when quality competition is supplied by a relatively high number of CS firms.

Result 5. Except for highly concentrated industries (low n), suitably-chosen mixed industries are always welfare-superior to Pure-OS or Pure-CS industries. These mixed states are usually welfare-superior because they feature enough OS firms to efficiently share costs and enough CS firms to provide quality competition. However, Pure-OS industries can still be welfare-superior to Mixed-OS/CS states where industry concentration is high.

[15] There is also a miniscule region where OS-firms develop code and CS-development is driven to zero. The area is located near the mixed cases border.

[16] Pure-OS is superior in a small number of cases where γ is very close to one.

6.4 Stable Outcomes in Case of Free Entry

We now know which situations (Pure-OS, Pure-CS, Mixed OS/CS) provide the most welfare. Here, we explore the extent to which markets actually deliver these outcomes. Except for Baake and Wichmann (2004) and Schmidtke (2006), previous contributions start from the assumption that industry size is fixed (Casadesus-Masanell and Llanes, 2009; Llanes and de Elejalde, 2009; Henkel, 2006, e.g.).[17] Absent compelling empirical evidence, however, ignoring entry seems artificial. Furthermore, the proportion of OS-firms ($\omega = z/n$) and welfare implications of these models depend on initial assumptions about the number of firms. Leaving industry size a free parameter limits their predictive power.

We adopt a different approach. Specifically, we use our very general model to systematically identify and evaluate the specific combinations of incumbents (n) and the proportion of OS-firms (ω) that we expect to find in real markets. More specifically, we assume that industries will continue to evolve under entry until additional OS and CS can no longer earn a positive profit by entering the market.

Section 6.4.1 assumes that a Mixed-OS/CS industry already exists and asks which proportions of OS-firms generate stable market equilibria (i.e. , resist further entry by OS and CS-firms). Significantly, we find that these proportions are systematically different from the target proportions that would be needed to maximize welfare. Section 6.4.2 then shows how industries can be locked into Pure-OS and Pure-CS states so that the transition to welfare-improving mixed industries never occurs.

6.4.1 The OS:CS Ratio in Equilibrium

In order to be stable, an arbitrary mix of CS and OS firms must satisfy the following conditions: (a) incumbent OS firms a earn profit ≥ 0, (b) incumbent CS firms likewise earn a profit ≥ 0, and (c) would-be additional OS and CS firms cannot earn a profit ≥ 0 by entering the market. We start by analyzing the case in which the total number of firms is large. Because such

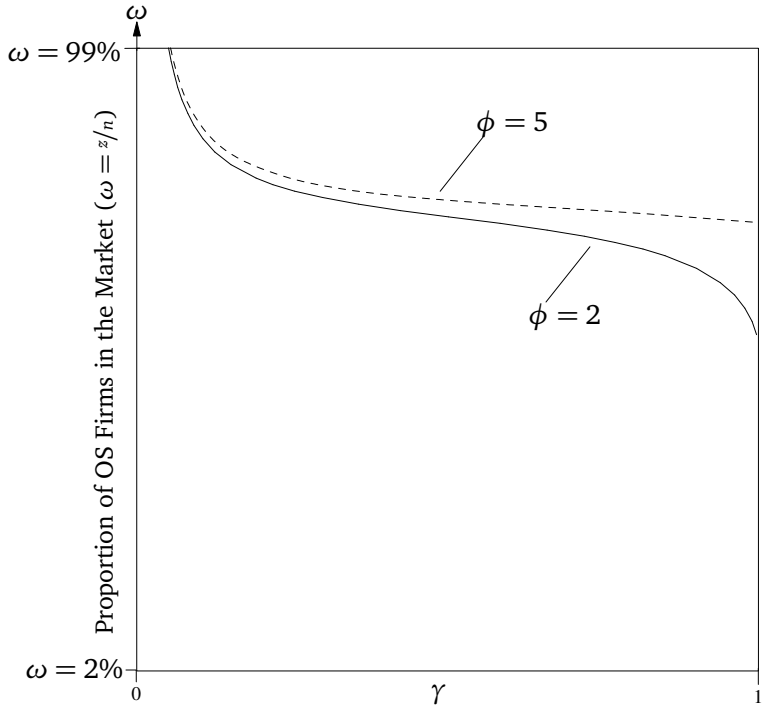
[17] Schmidtke (2006) likewise considers entry in the special case of duopoly incumbents.

markets are highly competitive, we expect each firm's (economic) profit to be identically zero. We thus make us of the zero-profits condition (5.16):

$$\pi_{i\in Z} = p_{i\in Z} \cdot q_{i\in Z} - c_{i\in Z} - C = \pi_{i\in R} = p_{i\in R} \cdot q_{i\in R} - c_{i\in R} - C = 0.$$

Significantly, the necessary condition $p_{i\in Z} \cdot q_{i\in Z} - c_{i\in Z} = p_{i\in R} \cdot q_{i\in R} - c_{i\in R}$ depends solely on γ, $omega$, n and ϕ . Figure 6.5 displays the (γ, ω) pairs needed to ensure stability for a market containing $n = 100$ firms where $\phi = 2$. (We solve the necessary condition, with $n = 100$, for γ and plot the results of $\phi = 2$ and $\phi = 5$. Taking into account that x^{os}, $x^{cs} \geq 0$ yields Figure 6.5.)

Figure 6.5: Entry-Resistant Proportion of OS Firms ($n = 100$)



Readers can confirm by inspection that the proportion of OS-firms (a) increases in ϕ, i.e. is higher where marginal costs increase steeply, and

(b) decreases in γ, i.e. declines in industries where 'stage two'-products are close substitutes so that competition is high. The relatively large proportion of OS firms for most parameters occurs because any arbitrary number of OS firms is almost always more profitable than an equivalent number of CS firms in our model. This result is dramatically different from Llanes and de Elejalde (2009), who find that OS firms are only more profitable than CS firms when cost-sharing dominates appropriability. We attribute this difference to quality cartel effects which systematically boosts OS profits in our model.

Despite this, CS firms still sell more bundles (i.e., have larger market share) than OS firms. Furthermore, as markets become more competitive (large γ), the average CS firm's market share grows while the average OS firm's market share shrinks. For $\gamma = 1$ the total number of bundles containing CS code greatly exceeds those containing OS code (see also Chapter 5). This result is strongly reminiscent of many industries (e.g. cell phones) where most consumers use products containing CS software even though OS firms greatly outnumber CS firms.

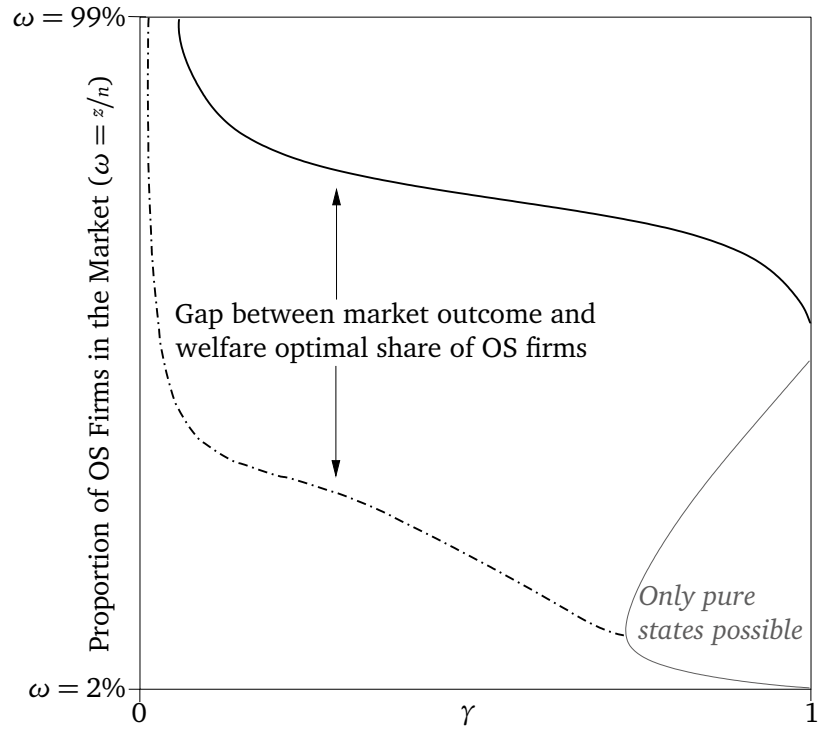
We have already seen that our model lets us calculate the welfare-optimal mix of OS and CS firms. Superimposing this plot on Figure 6.6 we see our stability condition (solid line) requires far more OS firms than the number needed to optimize welfare (dash-dotted line). (The market outcome is calculated as for Figure 6.5. To obtain the welfare optimal share of the $n = 100$, $\phi = 2$ case, we solve $\partial W/\partial \omega = 0$ for γ. Taking into account that x^{os}, $x^{cs} \geq 0$ and $\pi_{i \in Z}$, $\pi_{i \in R} = 0$ yields Figure 6.6.) This dramatic mismatch between the OS:CS ratio expected in equilibrium and the desired welfare-optimizing ratio is a central result of this article and poses an important challenge to policymakers.

The situation for concentrated (small n) industries is conceptually similar but more complicated. Since the number of CS and OS firms is always an integer,[18] the number of possible CS/OS ratios that can actually be realized in practice is small—for example, there are just four choices (0, $2/4$, $3/4$, 1) in a four-firm industry.[19] Thus, the stable OS:CS ratio predicted by our

[18] Of course this condition holds even for large n. Nevertheless, it can usually be neglected unless n is small.

[19] Since OS only generates cost savings when there are two or more firms to share code, the $1/3$ solution is economically indistinguishable from the case of three CS firms.

Figure 6.6: Mixed Industry with $n = 100$ - Welfare Optimality vs. Market Outcome ($\phi = 2$)



zero-profits analysis splits into two stable equilibria. For example in a four firm industry with $\gamma = 0.95$ our single predicted equilibrium at 0.69 can be realized by possible equilibria at 0.5 and also at 0.75. The 0.5 result improves welfare compared to our zero-profits calculation (0.69) while the 0.75 result makes it worse.

Result 6. Mixed OS/CS industries are only stable against entry when incumbents earn zero (or, for concentrated industries, very limited) profit. Using this criterion, we find that there are typically far more OS than CS firms in mixed industries. In particular, the number of OS firms is very much greater than the number required to maximize welfare. CS firms, on

the other hand, tend to have larger market shares than OS firms and this is especially true where firms' products are close substitutes (large γ).

6.4.2 Pure States and the Danger of Lock-In

We have seen that welfare optimality requires mixed OS/CS states. Suppose, though, that a particular industry starts off in a Pure OS- or CS-state. In this case, welfare improvements are bounded unless the industry can transition to a mixed state. Here, we explore the various circumstances under which industries can become locked in against such transitions.

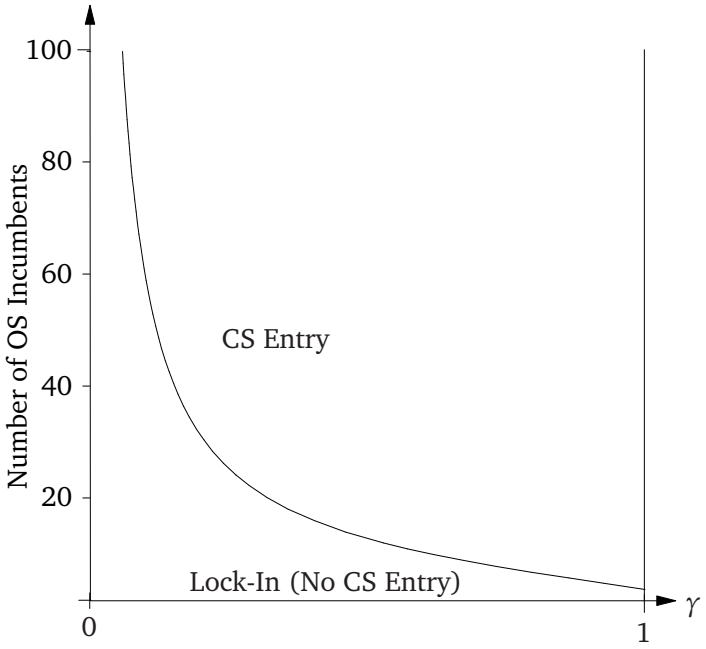
6.4.2.1 Pure OS

Suppose incumbents' profits in a Pure-OS industry are zero so that further OS entry is impossible. Our model allows us to calculate whether a CS entrant would earn a non-negative profit. Figure 6.7 displays this information for industries with different numbers of OS incumbents. (The zero profit condition for the z OS incumbents, $\pi_{i\in Z}^{n=z} = 0$, yields the corresponding stage two costs $C^{n=z} = p_{i\in Z}q_{i\in Z} - c_{i\in Z}$. CS-entry occurs only if the entrant's profits $\pi_{i\in R}^{n=z+1} = p_{i\in R}q_{i\in R} - c_{i\in R} - C^{n=z}$ are greater than zero. Solving $\pi_{i\in R}^{n=z+1} = 0$ for γ yields the boundary plotted in Figure 6.7.) This figure shows that industries can indeed be locked into a Pure-OS state that suppresses welfare. This only happens, however, where substitutability and/or the number of OS incumbents is small.

6.4.2.2 Pure CS

The case for OS firms' entry into a CS market is more subtle. Consider a Pure-CS industry in which incumbents' profits are zero. As before, we assume that this Pure-CS case is unstable if OS firms can earn a non-negative profit by entering. Since OS only confers economic benefits when firms are able to share costs, however, two or more firms must enter the market to reach a stable outcome. We therefore consider the case where two OS companies are (a) willing to enter the market, and (b) would earn a profit

Figure 6.7: OS Lock-In ($\phi = 2$)



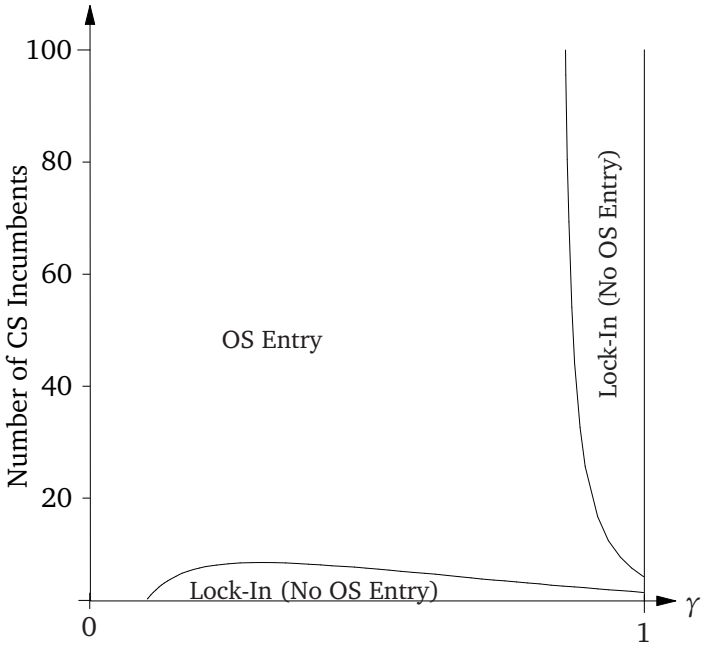
if both did so.[20] Then the first OS firm will not enter the market unless it knows that the second firm will also enter immediately afterward. This means, by induction, that it will enter if and only if the second firm is assured of earning a profit. But this is exactly what the GPL license does. By making its own code GPL, the first entrant commits to the specific OS regime that allows the second entrant to earn a profit. We therefore conclude that there is no fundamental reason why Pure-CS states cannot transition to Mixed-OS/CS states so long as GPL-like commitment strategies exist. (For an alternative motivation of OS-entry see Appendix D.5)

The stability of the Pure-CS state depends on whether the two OS entrants can earn non-negative profits. Figure 6.8 depicts this situation. (The zero profit condition for the r CS incumbents, $\pi_{i\in R}^{n=r} = 0$, yields the corre-

[20] Our argument does conceptually not depend on this assumption and holds equally for "two or more" OS-entrants.

sponding stage two costs $C^{n=r} = p_{i\in R}q_{i\in R} - c_{i\in R}$. OS-entry occurs only if the profits of each of the two entrants $\pi_{i\in Z}^{n=r+2} = p_{i\in Z}q_{i\in Z} - c_{i\in Z} - C^{n=r}$ are greater than zero. Solving $\pi_{i\in Z}^{n=r+2} = 0$ for γ yields the boundary plotted in Figure 6.8.) While the Pure-CS state is unstable for most parameters, it is stable for (a) the concentrated, low n industries, most characteristic of Silicon Valley, and (b) industries whose products that are close substitutes.

Figure 6.8: CS Lock-In ($\phi = 2$)



6.4.2.3 Strategic CS Adoption by Incumbents

The preceding analyses make no assumptions about why a particular industry should find itself in a Pure-OS or Pure-CS state. Here we show that firms in some concentrated industries can deliberately stabilize their industry in a Pure-CS state. Readers should note that this strategy is only available to

incumbents operating inside the "No OS-Entry" regions of Figure 6.8. Outside these regions, OS firms can enter no matter what incumbents decide.

We explore this phenomenon by inserting a new stage 0 into our model. In this extension, incumbents decide whether to adopt OS or CS in stage 0a and e entrants decide whether or not to enter in stage 0b. The model then proceeds as before through stage one (incumbents and entrants calculate how much OS and CS software to develop) and stage two (firms develop a complementary product at cost C and decide how much to supply). As before, we define equilibria by the condition that new entrants would earn negative profit. In general, we expect incumbents to strategically adopt CS over OS whenever doing so will (a) block OS entrants, (b) block additional CS entrants, and (c) produce greater profits than incumbents would earn in a world where open source entry occurred.

To see how this extended model works, consider the example of a highly concentrated ($n = 4$) and differentiated ($\gamma = 0.3$) industry in the case where $C = 0.1055$. Absent entry, firms can earn profits of about 0.062 by adopting OS and nearly 0.023 by adopting CS. However, the OS profit is illusory since the savings from code-sharing would allow two additional OS firms to enter the market after which industry profits would fall close to zero ($\pi = 0.000446$). By contrast, adopting CS is stable since in this case any new combination of CS and/or OS entrants would earn negative profits. We therefore expect our four incumbents to deliberately block entry by adopting CS. And this will be true even though entry following an OS decision would create an $n = 6$ Pure-OS industry that offers higher welfare. As usual, we expect these entry effects to be greatest for industries that are highly concentrated (small n) or feature low substitutability (low γ).

Result 7. Industries that start in a Pure-OS (-CS) state can be stable against the CS (OS) entry that would be needed to achieve a welfare-superior Mixed-OS/CS state. Incumbents may also adopt CS strategically in cases where choosing OS would facilitate entry and erode oligopoly profits.

6.5 Government Intervention

So far, we have limited our analysis to asking how much software we expect OS and CS firms to supply in equilibrium. However, government can

also intervene to change this private sector outcome. For example, we have stressed the role of OS as a de facto cartel that suppresses quality competition. In principle, government could redress this using antitrust (competitions) policy to render OS collaborations illegal. This, however, would eliminate OS even where we expect it to be welfare-superior. More fundamentally, it seems wasteful to discard OS's cost-sharing advantage unnecessarily. This section asks whether traditional policy levers based on taxation and preferred procurement policies can provide a better solution.

6.5.1 Tax Policy

This section examines how government can use lump sum and progressive taxes and tax breaks to change industry output of OS and CS code.

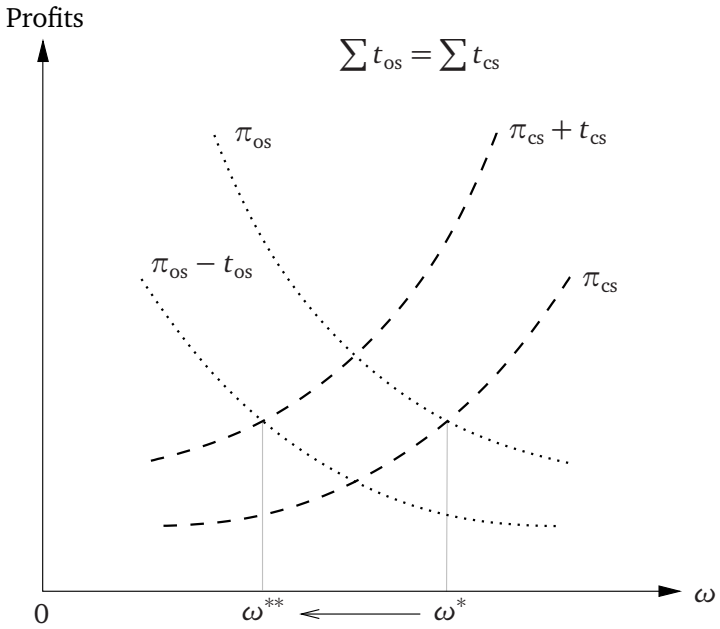
6.5.1.1 Discriminatory Lump-Sum Taxes

We have seen that market forces in Mixed-OS/CS industries invariably deliver far more OS firms than welfare-optimization requires. This suggests that the governments should view typical proposals to foster OS as an infant industry, say through special tax breaks, with suspicion. To the contrary: The most obvious way to improve welfare in our model is to promote CS firms instead. This is best done by taxing OS firms, giving tax breaks to CS firms, or both.

For concreteness, consider a scheme where government imposes a fixed, lump-sum tax on OS firms and uses the proceeds to give lump-sum tax breaks or subsidies to CS firms. Detailed calculation confirms the intuition that these interventions do indeed reduce the equilibrium ω in our model

Figure 6.9 presents a stylized illustration of this process. Here, suitable taxation adjusts the market outcome so that it coincides with the desired welfare-optimal proportion of OS firms. Furthermore, our hypothetical lump sum taxes and tax breaks transfer profits from OS to CS firms while leaving total industry profits unchanged. This means that both the desired welfare-optimal OS:CS ratio and achievable welfare remain the same as before. This suggests that taxes and tax breaks are a potentially powerful tool. In practice, the main drawback is that government may not be able to estimate the welfare optimal OS:CS ratio and/or proper taxation with any

Figure 6.9: Lump-Sum Tax on OS Firms and Tax-Breaks for CS Firms (Stylized Illustration)



degree of precision. In this case, ambitious transfer schemes could easily end up over-taxing OS firms and over-subsidizing their CS competitors. Additionally, per-firm profits after new taxes and tax breaks could sometimes be negative, forcing government to make up the difference from general revenues.

6.5.1.2 Progressive Input-Tax

Lump sum taxes are not the only option. In principle, government can also use progressive input-oriented taxation to change OS and CS-firms' effective marginal cost function to $(\phi + t)x$ where t denotes the tax. Detailed calculation shows, however, that this intervention does nothing to bring equilibrium OS:CS ratios in mixed industries closer to welfare optimality. The reason is that taxation simultaneously shifts both the OS:CS equilib-

rium ratio and the target welfare-optimal ratio in the same direction. This leaves the gap between the two just as wide as it was before.

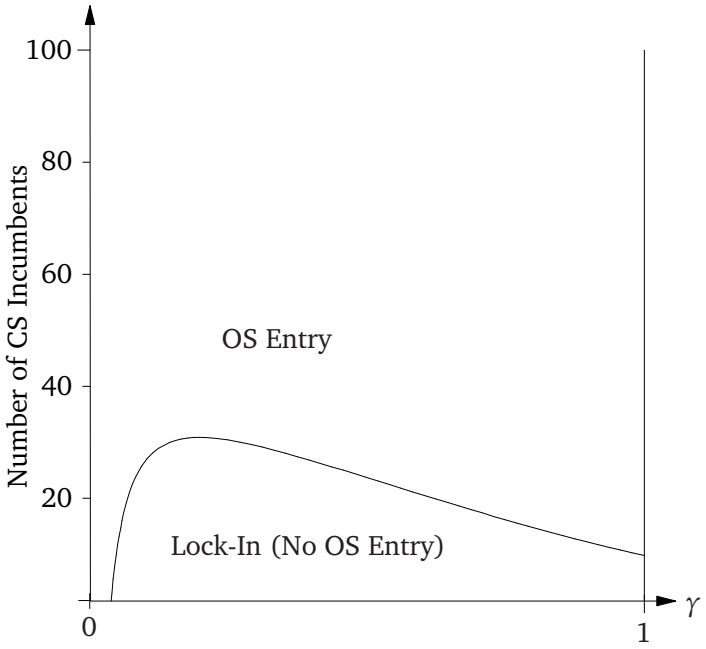
On the other hand, changes in (effective) marginal costs do affect entry. This suggests that they could have a significant impact in helping Pure-CS or Pure-OS industries evolve into Mixed-OS/CS industries. Even here, however, value-added taxes provide an ambiguous intervention. To see this, compare a government tax policy that increases the slope of the (effective) marginal costs to $\phi + t = 5$ (Figure 6.10) compared to the $\phi + t = 2$ case depicted Figure 6.8. On the one hand, taxation has eliminated the right-hand (high γ) region where CS industries were formerly stable. This clearly facilitates transitions to welfare-improving mixed states. On the other hand, taxation also increases the second region near the bottom of the figure where CS is stable. This suggests that policymakers trying to destabilize a Pure-CS industry could accidentally reinforce it instead. This is particularly likely to happen for high γ/low n industries where the two regions are close to one another.

6.5.2 Government Provision of OS Software

Government can also intervene by paying contractors to create additional OS code over and above what the market supplies. We have already remarked that many countries have established different institutes, collaborations, grants, and partnerships to fund OS software development and adoption. The analysis is simplest for pure-CS markets where incumbents would otherwise be able to block entry by OS firms. Here, government-supplied OS reduces entry costs so that an OS sector can establish itself.

The case for government funding in mixed industries is more ambiguous because it takes place at two levels. The first effect takes place entirely within the private sector. Government-supported OS simultaneously (a) makes OS firms more profitable so that the equilibrium ω increases, and (b) reduces OS firms' incentives to produce their own code (government OS crowds out private investment) so that the welfare-optimal ω falls. The net result is to make the already-large mismatch between the equilibrium and desired welfare-optimizing proportion of firms even worse. Indeed, sufficiently large government OS investments can drive CS firms out of the market entirely.

Figure 6.10: CS Lock-In in Case of $\phi + t = 5$



The second effect takes place across the entire economy. Suppose that government only cares about achieving the correct level of production without regard to how much the private sector contributes. Suppose further that government contractors have the same cost structure that the private OS firms do. Then the new, government-supplied code writers can be thought of as an additional OS firm that chooses output based on government fiat instead of our Cournot analysis. This allows policymakers to select any desired level of quality and still benefit from significant sharing with other private sector OS firms.[21] In principle, government can use this lever to increase welfare although it may be hard to estimate how much OS software to fund in practice.

[21] Llanes and de Elejalde (2009) similarly predict that government investment in OS increases the OS stock directly and also encourages more OS firms to enter the market.

6.5.3 Government Procurement Preferences

We have already noted that government is a major software purchaser. Many governments have tried to turn this spending into a policy instrument by establishing formal preferences (and even mandatory requirements) that systematically favor OS over CS products. This provides a powerful incentive for new OS entrants in cases where Pure-CS industries that would not otherwise evolve into welfare-improving mixed states.

In order to analyze the impact of preferences on mixed industries we add government demand to our model by assuming that government procures D bundles. We start with a neutral government as reference: Absent preferences, government buys one bundle from each firm so that each firm's demand is shifted by $d = D/n$. With government OS preferences, however, each OS firm sees an extra demand of $d = D/z$ while CS firms see d = 0. As with government provision of OS code, this widens the gap between the equilibrium ω and the ω needed to maximize welfare. Furthermore, the equilibrium proportion of OS firms increases. Unlike the case of government provided OS, this intervention reduces welfare. The new equilibrium ω has lower welfare than the equilibrium ω for a neutral government. Conversely, government procurement preferences for CS[22] yield the opposite results and increase welfare.

Result 8. Government can use various policy instruments to improve welfare by reducing high equilibrium proportions of OS firms. These include competitions policy (antitrust), taxation and tax break schemes, government-funded OS development, and government procurement preferences. Of these, tax policy provides the most natural instrument for achieving the target proportion needed to optimize welfare through private sector investment. By contrast, government-provided OS actually increases the gap between the desired and actual OS:CS ratio and depresses private OS investment still further. Despite this, government-funded OS still improves social welfare by boosting the total (private + government) supply of OS. Government procurement preferences for OS software have the worst outcome by not only increasing the gap between desired and equilibrium OS:CS ratio but also reducing total welfare.

[22] In this case each CS firm has an extra demand of $d = D/r$ and CS firms have $d = 0$.

6.6 Discussion

We have presented a general model that allows us to compare equilibrium OS:CS ratios against the target ratios that would be needed to maximize welfare. However, it remains possible that real markets may require more complex models. This section explores the extent to which such extended models would qualitatively change our results.

6.6.1 Basic Analysis

Our model OS output reflects a delicate balance between (a) lower per-firm costs through shared development, (b) reduced appropriability and hence smaller investment incentives in case of strong quantity competition, and (c) a cartel effect that suppresses quality competition among OS firms. The first two effects are fairly straightforward and have previously been noted in, for example, Llanes and de Elejalde (2009), and Henkel (2006).

To the best of our knowledge, however, our third ("quality cartel") factor has never been noticed before. This is puzzling because Llanes and de Elejalde (2009)'s model is somewhat similar to ours. We conjecture that quality cartels do not appear in their model because they require firms to set quality and prices in a single, simultaneous decision. Introducing a single simultaneous decision similarly suppresses quality competition (and therefore quality cartel effects) in our model.[23] We therefore expect a similar "quality cartel" effect to appear in an extended Llanes and de Eleljalde model—and, indeed, most generic models that require firms to make their quality and price decisions sequentially. Such models are also more realistic given the substantial lags between most firms' R&D (i.e. software) investment and pricing/output decisions.

[23] We analyzed a one-stage version of our model where firms choose the profit-maximizing (price, quality) pair. In the Pure-CS case quality competition is so weak that output continues to decline even at high γ.. In the Pure-OS case, our results approximate the results for a formal OS quality-cartel described in Appendix D.1

6.6.2 Technology Assumptions

The quality of our bundles is completely determined by a single technology (software) which is (a) indivisible, (b) can be developed jointly, (c) can be shared, and (d) confers an identical quality "boost" on every product with which it is compatible.[24] In principle, all of these assumptions can be relaxed.

Our first assumption that all quality comes from a single indivisible technology ("software") is more general than it looks since "software" can trivially be relabeled (a) to include other technologies (e.g. hardware) and (b) to exclude any technology (including GPL code) that is only usable by the author. Assuming that software is indivisible, however, does exclude situations where quality depends on multiple technologies each of which can be separately developed using OS or CS methods. One natural speculation is to ask what happens when firms can also invest in a second technology whose benefits are primarily limited to their own 'stage two'-product. Intuitively, we would expect this additional quality investment to drain resources from stage one R&D leading to fewer OS firms, less cost-sharing, and less development of shared software. This is more or less what happens when Llanes and de Eleljalde (2009) allow firms in their model to invest in a second technology focused narrowly on their products. The existence of a second, severable technology also facilitates strategic behaviors in which firms keep at least one technology closed as a barrier to entry (Schmidtke 2006).

Relaxing the second assumption that firms can develop code jointly is likely to be strongly model-dependent and should probably await convincing evidence that such failings actually exist. In the meantime, we note that Henkel (2006) has explored a model in which joint development is impractical at the level of individual OS modules so that each project is effectively controlled by one (and only one) company. Henkel argues that firms self-select toward developing whichever modules they value most and that this biases total OS investment upward. More generally, one can also imagine models in which OS joint development is possible but inefficient or imperfectly monitored. This could happen, for example, if participants

[24] Conversely, the software confers no benefit on incompatible products. These are automatically excluded from out model.

adopted mixed strategies that encouraged them to strategically withhold effort from the collaboration in hopes that some other member would do the work (Johnson, 2002)

The third assumption that firms can share OS might be relaxed if, for example, substantial "tacit knowledge" was needed to use completed software. We therefore explored a variation of our model hat features a "spillover parameter" $\sigma \in]0,1]$, such that $\alpha_i = 1 + \sigma X^{os}$. We find that our OS results gradually converge to CS where spillovers are small.

Finally, completed software may not boost all firms' products quality equally. Naively, we would expect the presence of some firms that gain relatively little from OS to produce free-rider effects. However, Henkel (2006) has shown that the existence of specialized interests within firms can actually increase total OS production. Relatedly, firms' willingness to invest in OS could depend on the size of their respective stage two markets. It would be natural to investigate this by allowing different qualities of the 'stage two'-products in our model. For now, it is probably safe to say that the answer will sensitively depend on how many separate technologies exist and the distribution of preferences among firms. Absent detailed empirical guidance, it will be hard to know which models to investigate.

6.6.3 Demand Side Assumptions

Our model assumes that consumers choose between products based quantity supplied, substitutability, and a one-dimensional 'quality' parameter based on the amount of software produced. We recognize, however, that consumers may have idiosyncratic preferences for particular products. Naively, we expect strong consumer preferences to reduce the payoffs from quality improvements leading to lower code production. To the best of our knowledge, Llanes and de Elejalde (2009) are the only authors who have investigated firms' decisions to invest in quality using a Hotelling model that includes for idiosyncratic demand and allows for $n > 2$ firms. While their results are broadly similar to ours, Pure-OS industries are indeed much more common in their model. We conjecture that idiosyncratic consumer preferences reduce the importance of appropriability so that shared OS software production becomes more lucrative.

Similarly, the degree of substitutability (γ) is exogenous in our model. Over time, however, one might expect firms to design new products strategically so that γ becomes endogenous. This could be accomplished by, for example, linking our two appropriability variables n and γ. Alternatively, one might think that shared code would make OS products more similar (higher γ) to each other than to CS products. Llanes and de Elejalde (2009) explore this possibility by introducing different substitutability parameters for bundles that contain OS compared to bundles that contain CS. Not surprisingly, they find that increased substitutability leads to greater competition among OS firms which, in turn, makes CS firms larger and more profitable. We conjecture that endogenizing substitutability in our model would produce similar results.

Finally, we have limited our analysis to the case where products are substitutes. However, not all products compete and some are complements. As Schmidtke implicitly points out, OS provides a natural way for firms to encourage the production of complements that promise to increase their own product sales. Extending the current model to include this case would be reasonably straightforward. Doing so would probably mitigate the free rider but not the cartel effect. Furthermore, the number of such complementary products—and hence the importance of Schmidtke's observation—remains unclear.

6.6.4 'Spooky' OS Incentives

We started this chapter by remarking that commercial incentives have increasingly crowded out other incentives. However, volunteer labor remains important for many OS collaborations and dominate some. In principle, some of this voluntarism may reflect the desire for future wages and could in principle be endogenized in our model as a kind of prize. In general, however, many motives (reputation, altruism, fugue-state) are likely to remain forever outside predictive modeling. Following Einstein's reference to "spooky action at a distance" (Pais, 1983) we refer to such non-commercial incentives as 'spooky OS' in what follows.

The effects of spooky OS on our model are similar to those already considered for government provided OS. Thus, we expect increased spooky OS to (a) increase the total supply of OS, (b) increase the total number

of OS firms, and (c) reduce the total amount of OS produced for commercial reasons. The main difference is that spooky OS potentially increases welfare faster than government OS. The reason is that programmers who gain psychic benefits from voluntarily supplying spooky OS have already been compensated. By definition, such hobby activities have no opportunity costs and thus increase welfare even more (see also von Engelhardt and Pasche 2004).

6.7 Conclusions and Outlook

Today's open source is increasingly dominated by business strategies in which firms make proprietary products whose quality depends on a shared OS code base. We have presented a generic Cournot model based on simple, realistic assumptions about the costs of developing OS and CS code. We find that Pure-OS industries are welfare-superior to Pure-CS states so long as quantity competition is modest so that appropriability is high. Otherwise, Pure-CS industries are welfare-superior except for some industries where cost functions rise so steeply that CS firms are forced deep into diminishing returns in the case of nearly-identical products and strong quality competition. Finally, mixed industries containing a suitably-selected proportion of OS firms are superior to both Pure-OS and Pure-CS industries. This is because the presence of CS firms introduces quality competition which induces OS firms to produce much more code than they otherwise would. Ironically, then, OS is only able to realize the full benefits of cost-sharing when CS firms are present.

Unfortunately, we find no evidence that the OS:CS ratios needed to maximize welfare are ever realized in practice. Instead, equilibrium mixed states consistently produce too many OS firms. Additionally, many pure markets are stable and cannot transition to the mixed markets needed to improve welfare. In some cases, incumbents can also deliberately block entry by choosing CS.

Policymakers trying to improve welfare can cope with these problems in several ways. Probably the most elegant (and also revenue-neutral) proposal is to tax OS firms and use the proceeds to subsidize their CS rivals. A more politically-correct—and ambitious—alternative would be for gov-

ernment to fund OS production directly. This would increase welfare over the market outcome although it also widens the gap between the equilibrium and welfare optimal proportion of OS firms. By contrast, procurement preferences that concentrate government spending on OS bundles should be rejected. Such schemes invariable decrease welfare and make the mismatch between the equilibrium and welfare-optimal proportion of OS firms even worse.

Finally, Section 6.4 shows that inferior lock-ins in Pure industries are possible, and incumbents might choose CS over OS to block entry. This points to the importance of entry. In addition, firms in the model are symmetric, except for the horizontal and vertical (quality) differences of their products. Chapter 4 points out that different individual resources are one reason why some economic agents choose the OS regime and others the CS regime. Taking these two aspects together leads to the question of how firms entering the market with OSS-based business models differ from those with CSS-based business models. The next chapter deals with this question.

7 Who Starts with Open Source? Institutional Choice of German ICT Start-Ups

7.1 Introduction*

Institutions affect start-up activities and entrepreneurship in many ways.[1] Typically, institutions differ among regions, countries or sectors. Research on the connection between entrepreneurship and IPRs (Burke and Fraser, 2007), or between entrepreneurship and institutions in general (Amoros, 2009; Hall and Sobel, 2008; Nyström, 2008) therefore has to compare regions or countries. A more direct comparative research is possible only if firms from the same country and sector can choose different institutional (sub-)settings. The ICT sector offers such an opportunity, as here firms can choose between the two different, software-related IPR regimes OSS and CSS. Firms with OSS- and CSS-based business act not only in the same sector but often directly compete in the same market. This chapter analyses the choice of OSS- vs. CSS-based business models of new business formations in the German ICT sector.

As already mentioned, OSS vs. CSS lead to different allocations of IPRs and different governance structures regarding the software development process. This has implications for firms using OSS- rather than CSS-based business models. The following is a brief description of these differences, for more details see 1.2.2, 1.2.3 and Chapter 4. CSS is based on exclusive ownership of the software and its source code. Beside cases like contract programming, CSS-users do not have the right to change or further develop

*This Chapter is based on Fritsch and von Engelhardt (2010)

[1] See for example Acs et al. (2008); Foss and Foss (2006); Henrekson and Sanandaji (2010); Henrekson (2007), for an overview see Bosma et al. (2010).

the code (abusus). CSS-licenses only transfer the right to use the software as it is (usus and usus fructus). Consequently, the source code is "closed" as customers receive only the binary code, which is the machine-readable version. CSS-users have to pay license fees for using the software, while the source code remains with the developing firm. Hence, the exclusively owned source code is an asset for the developing firm. In contrast, OSS is based on inclusive ownership. The OSS licenses transfer the usus, usus fructus and also the abusus rights. Therefore OSS is marked by free access to the source code in order to enable users to change and further develop the code. However, many OSS licenses contain restrictions to ensure that OSS cannot be turned into CSS. Hence, firms cannot use OSS code as an exclusive asset, and one cannot separate producers from users via IPR-allocation. Thus, OSS is developed by community-based projects. Such OSS projects are governed by a mix of formal and informal institutions. This implies, for example, that firms with OSS-based business models have to act compliant with the community rules, otherwise they risk that cooperation is stopped etc. Finally, firms with OSS business models have to generate revenue by selling products (goods or services) that are complements to the free accessible OSS.

So far, research on OSS-based business models has focused on OSS firms only. The literature analyzes the various business models of OSS-firms (Bonaccorsi et al., 2006; Ghosh et al., 2002a), why and how they are engaged in the OSS community (Dahlander and Magnusson, 2006; Dahlander and Wallin, 2006; Rossi and Bonaccorsi, 2006) or release software products under OSS licenses (Fosfuri et al., 2008; Harison and Cowan, 2004), and whether such community-participation has an impact on their performance (Stam, 2009), etc. The relationship between OSS and *entrepreneurship* is analyzed by Gruber and Henkel (2006), who focus on new ventures that apply embedded Linux. Based on data from personal interviews they conclude that the key challenges for new ventures discussed in the entrepreneurship literature are of less relevance for such OSS-firms. To the best of our knowledge, the only two comparative studies of OSS and CSS firms are Lamastra Rossi (2009) and Harison and Koski (2010). Based on a sample of 134 software solutions developed by Italian small and medium sized enterprises, Lamastra Rossi (2009) concludes that OSS solutions seems to be more innovative. Harison and Koski (2010) use survey

data from 170 Finish software companies and analyze how the firms' properties shape their OSS-vs-CSS decision. They distinguish between firms with no OSS (firms that provide only CSS) and firms with OSS (either purely or as hybrid strategy, i.e. an OSS-CSS-mix). Harison and Koski (2010) find that the decision to use some vs. no OSS can be explained by several characteristics of the software firms. In particular, human capital (education) has a positive impact on OSS strategies, and firms that are younger and smaller more often apply OSS supply strategies. Software firms owned by one or two individuals or a family tend to be CSS-only. Finally the magnitude of the service variety provided by the firms has a positive impact on the propensity to adopt OSS strategies.

The study presented in this chapter analyzes which aspects shape the institutional choice (OSS- vs. CSS-based business models) of German ICT start-ups. We distinguish different levels of OSS-intensity—ranging from never OSS to always OSS—in the different business fields of the ICT start-ups, ranging form hardware and software to web-related services. The remainder of this chapter is organized as follows. Section 7.2 gives an overview on the relationship between institutions and entrepreneurship based on a conceptual model. We then develop in Section 7.3 our hypotheses about the connection between the characteristics of new businesses and their decisions to use OSS- or CSS-based business models. Section 7.4 introduces the data and Section 7.5 reports the results of the empirical analysis. Finally, we discuss our findings and draw conclusions for further research (Section 7.6).

7.2 Institutions and Entrepreneurship

The relationship between institutions and entrepreneurship can be explained on the basis of a conceptual model (Figure 7.1). The starting point of this conceptual model entails the feasible entrepreneurial opportunities, i.e. existing chances that are in principle accessible for everyone (for a more detailed exposition see Bosma et al., 2010). The available entrepreneurial opportunities are in many respects shaped by the governing formal and

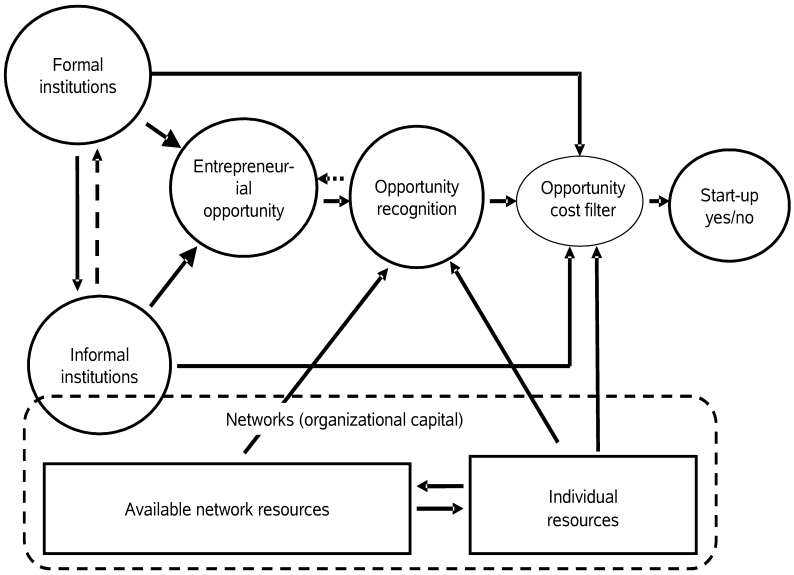
informal institutions.[2] Examples of how formal institutions shape the entrepreneurial opportunities are the formal requirements that have to be fulfilled before legally starting a business, or labor market regulations. Informal institutions like certain modes of conduct, routines, or even culture (Freytag and Thurik, 2010; Nguyen et al., 2009), also determine (the perception of) entrepreneurial opportunities. The differences of OSS and CSS with respect to formal (licenses) and informal (e.g. the hacker ethics) institutions thus determine differences in the entrepreneurial opportunity of OSS- vs. CSS-based business models. The licenses and the resulting IPR allocation determine the availability of code, and how this code can be used. For example, an OSS-based business model can be more flexible as the start-up firm has access to the source code and can thus change the given code (e.g. customize)[3] Furthermore, the informal institutions of the OSS community, for example the culture of helping each other but also the notion of `contributing back' (expected reciprocity), shape the entrepreneurial opportunity.

Because the informal institutions, the unwritten rules, emerge through the networks of interactions and face-to-face contact, these networks can be regarded as part of the informal institutions. The overlap between networks and informal institutions is particularly due to the fact that certain rules or a certain 'culture' may be specific to a certain network and do not pertain to other networks, like the example of the OSS hacker ethics shows.

[2] There is a pronounced interdependence between the formal and the informal institutions. The formal institutions have often emerged from the informal institutions. However, the governing formal institutions feed back to the informal institutions by providing the legal framework for interaction that may lead to the further development of formal rules. The emergence of OSS is a good example for such a development. OSS emerged from the dissatisfaction with CSS. Based on the informal institutions of hacker ethics and a culture of making software freely available, Richard Stallman designed the GPL. With the GPL Stallman invented a new concept of copyright-based ownership. It was an act of institutional entrepreneurship that changed the level of institutionalization by transferring some cultural norms—i.e. informal institutions—into formal institution, i.e. the GPL

[3] Of course, customizing given CSS code is also possible. But the difference is that in case of CSS the firm needs a special license agreement with the original CSS developer and has to pay license fees. In case of OSS the code can be further developed without such a special agreement.

Figure 7.1: Institutions and Entrepreneurship: A Conceptual Model



(Source: Bosma et al., 2010)

The realization of entrepreneurial opportunities depends on the individual recognition of these chances as well as on the opportunity costs of starting a business instead of pursuing available alternatives (e.g. dependent employment, unemployment benefits and education opportunities). The opportunity costs work as a 'filter' for the realization of business concepts. According to the model of occupational choice (Knight, 1921; Lucas, 1978; Kihlstrom and Laffont, 1979; Holmes and Schmitz, 1990) benefits that they anticipate to earn through employment with those they expect to accrue from starting a business. The opportunity costs of entrepreneurship are affected (a) by some formal institutions like unemployment benefits or the tax system, and some informal institutions like the social prestige of self-employment ot the family history, and (b) by the individual resources as well as by the available network resources.

Individual resources comprise all those input factors needed to start a business such as labor, human capital, financial capital, organizational capital, social capital, physical capital, knowledge and technology. These relevant resources also include personal characteristics such as education, risk attitudes, motivations and preferences, that are partially innate, but that are also influenced and shaped by a persons' environment (e.g., the role models).

A persons' network of relationships, his ego network, represents his or her organizational capital, i.e. the ability to mobilize resources that are owned by others for a venture. An important distinction has to be made between the overall network and the ego network of an individual actor. Networks may particularly enable actors to connect their individual resources to the resources of other actors, thereby aggregating and transforming these resources into the overall resource-stock. The total available resources are the aggregate of the actor's individual resources and the resources possessed by other actors. They shape an actor's ability to recognize and to pursue entrepreneurial opportunities. The OSS 'community' is an example for such a type of network that should reduce the amount of individual resources which are needed to start an own businesses in this sector.

7.3 CSS- vs. OSS-Based Start-Ups: Hypotheses

ICT-Firms have an institutional choice, as they can choose between OSS- and CSS-based business models. OSS and CSS co-exist in the market as well as within firms (hybrid business models). As already mentioned, OSS business models are based on selling complementary products like hardware, premium versions of the software, or different kinds of service including individually customized OSS. Based on the conceptual model of institutions and entrepreneurship, we will now derive hypothesis about the factors that shape the degree of OSS vs. CSS usage of ICT start-ups.

First, the decision to use OSS or CSS has implications for the individual resources needed to start up: CSS-based start-ups either have to pay license fees for the software, or have to develop the code on their own, i.e. by own staff. OSS-based start-ups on the contrary can use the code developed by

the community and may be supported by other member of the community in the development of the software code. The support that OSS-based start-ups may receive from the OSS community is mainly based on two channels: collaborative code development and help via the respective discussion forums. Particularly the support received via discussion forums, which is an important part of the OSS culture, may also reduce the amount of personnel that is needed to start a firm. Thus the available network resources of an OSS-based strategy can substitute for individual resources. Consequently, OSS-intensive start-ups should need less personal resources to be competitive than CSS-based start-ups. Hence, we assume that

H1: OSS-based new businesses start with a smaller size than CSS-based start-ups.

The expected statistical relationship between OSS usage and start-up size raises the question of causality: Do founders choose OSS-based business concepts out of need because they have not enough resources available to set-up a CSS-based firm, or is the choice of an OSS-based concept primarily motivated by other reasons which are independent of resource requirements? Regarding capital we can control for this issue to some degree based on information whether a lack of capital was one of the main problems when starting the business. If we find that OSS founders have indeed experienced stronger capital bottlenecks than founders of CSS-based firms this can be regarded an indication that lower capital requirements of an OSS-based concept may have been the reason to favor this type of strategy. If capital bottlenecks play no special role for OSS founders, then the smallness of OSS start-ups may be regarded more a result than a reason to choose OSS. We will assume the latter, as anecdotal evidence indicates that OSS facilitates new entrepreneurial opportunities and/or it's recognition. Therefore we expect that

H2: OSS-intensive start-ups less often have the problem of a lack of own capital than CSS-based new businesses.

One characteristic of OSS-start-ups in the ICT sector could be having more problems to convince potential financiers. Basically there are two reasons why OSS-based founders may face special problems in this respect.

First, there might be a lack of acceptance because OSS-business models as such are relatively new and may be regarded more skeptically by potential investors than the more conventional CSS-based concepts. Moreover, financiers might expect that potential customers will only hardly accept OSS-based solutions. Second, financiers might assess OSS-based start-ups more risky since the OSS-code as such is public and can, therefore, hardly be sold. Thus, the economic success of OSS start-ups relies more on complementary products and services than is the case for CSS start-ups. Moreover, OSS-based firms depend on the respective software community. Firms have to act compliant with the informal rules of the community otherwise they will be cut off further cooperation. Generally, the future of an OSS-based business models depends on the future of the collaboratively developed OSS project. It may be split up (forking) or even die out because of lack of further voluntary contributions. In other words: The differences in (formal and informal) institutions of OSS vs. CSS determine differences in the entrepreneurial opportunity of OSS-based business models. Potential financiers, who have a different opportunity cost filter than the entrepreneurs, may thus valuate the OSS business models lower than CSS-based ones. For these reasons, we expect that

H3: OSS-based start-ups have more problems of convincing potential financiers than CSS-based new businesses.

As already mentioned above, OSS vs. CSS affects the opportunity cost filter as it lowers the barriers to entry in terms of individual resources needed. But relatively low entry barriers for OSS-based businesses may attract mainly start-ups with relatively poor quality in terms of qualification and experience which would be hardly able to set up a CSS-based firm (Fritsch and Schroeter, 2009; Parker, 2009). This does in no way mean that all OSS-start-ups are of low quality, but low quality start-ups are more likely to be OSS-start-ups than be based on CSS. With regard to the qualification of the personnel one may, therefore, expect that OSS start-ups with less experienced founders and less educated staff to have a higher level of OSS-intensity.

H4: Founders and personnel of OSS-based start-ups have lower levels of qualification and experience than CSS-based new businesses.

Please note that Hypothesis 4 is not contradictory to Harison and Koski (2010) who find that software firms with OSS-business models have higher human capital (education). Harison and Koski's statement relates to established software firms, while we relate to start-ups (of the whole ICT sector). For example, low qualified start-ups may have a lower survival rate, and established OSS-firms can have started with CSS.

Furthermore, due to the lower entry barriers for OSS-based firms we expect that

H5: Founders of OSS firms are more likely to be necessity motivated than persons which set-up CSS-based firms.

OSS-based businesses have lowers barriers to entry, what makes it relatively easy to realize a business idea (opportunity) as well as to establish a firm out of necessity. The reason to expect more necessity-motivated founders among the OSS firms is that opportunity-based start-ups are more likely to enter also in fields with relatively high entry barriers.

7.4 The Data

Our data is based on a survey among founders of ICT firms in Germany conducted in Fall 2009. In a first step we sent out a postal invitation letter to about 6,000 firms[4] asking to participate in an online-inquiry containing

[4]The addresses of the ICT firms have been selected from the heise IT-Markt, which is an online catalogue for German ICT firms run by the heise publishing company. Among other products heise publishes the periodical c't, a highly reputated IT- journal as well as the German version of MIT's Technology Review. The homepage of heise is a well known web-adress for ICT issues, and with the heise news ticker the company runs one of the most successful (German) ICT news portals. The heise IT-Markt offers German ICT firms the opportunity include their profile, i.e. their name, address, product-portfolio etc., into a freely available internet-data base. Potential customers can search for ICT firms in this data base using different search parameters (region, products, etc.). Such a platform is a unique opportunity for being recognized by customers, especially for small and medium sized firms. Starting by the end of March 2009, we collected names and postal addresses of firms operating in the industry sub-categories which were of interest for the purpose of this study. After cleaning the data from duplicates and misleading entries we ended up with addresses of 15,300 firms. From this database we drew a random sample of 6,000 firms.

an individual access key. After about two weeks we sent out reminder. As a result, more than 700 founders of ICT firms filled out our online-survey completely. As some of those firms did not sufficiently match the focus of our research we finally ended up with a dataset of usable answers of 680 founders which makes a response rate of more than 11 %.

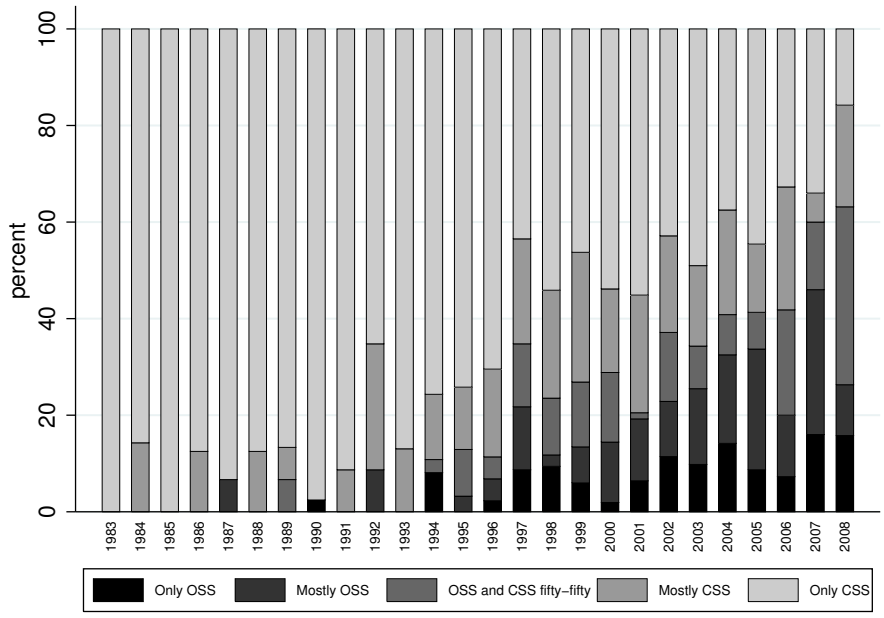
The survey raised some general information about the firm (e.g., number of employees, date of start-up, problems at start-up etc.) as well as about the use of OSS and CSS. The questions focused particularly on OSS- vs. CSS-based business models, thus tried to investigate in how far OSS is part of the end-product that is sold to the customers (OSS-business model). This does not included aspects like the use of freely available software such as OpenOffice for business correspondence etc. Participants in the survey were asked to select their activities from a predefined set of business fields. If a firm was active in more than three such business fields we asked to select the three most important ones in terms of revenue. Specific questions where then asked for each of these fields. The firm-founders were also asked to select those up to three business fields they were active in when founding the firm. For each of these business fields at the time of start-up we then asked for the usage of OSS[5]. Table E.4 and E.5 in the Appendix provide an overview of the information we use in the current chapter.

Over the years, more and more firms got engaged in OSS business. This institutional change in the ICT sector, induced by institutional choice (OSS- vs. CSS-based business models), is also reflected in our data. In Figure 7.2 the reader can see the shift in the institutional choice of founders: In the period 1984-2008[6] the share of start-ups with no OSS, hence CSS-only-

[5]Regarding OSS we asked whether the software used in the respective business field at time of founding was OSS. Possible answers were "yes, exclusively", "mainly", "about 50%", "a small extent", "no (nearly) never", and "I do not want to answer this question".

[6]Start-ups before 1983/84 are not of interest here. The reason is the the history of OSS vs. CSS (see also 1.2.1): The concept of CSS as we know it today evolved not earlier than the late 1970s and early 1980s. As a reaction, Richard Stallmann announced the plan for the GNU project on September 27, 1983. On January 5, 1984 software development of the GNU project began. Thus the year 1984 can be seen as the initial year for what we nowadays call Open Source Software. (Two other possible cut-off points are the year 1986, as the Free Software Foundation published the free software definition in February 1986, and the year 1989, when the first version of the GPL (the GNU General Public License) was published.) Thus, before 1984 it does not really make sense to

Figure 7.2: Institutional Change by Institutional Choice: OSS/CSS Usage in Business Fields of German ICT Start-Ups 1983 to 2008



start-ups, considerably decreased. Since the late 80s there is a rising share of start-ups (their respective business areas) with a level of OSS usage of 50% and more. At the same time, newly started businesses seem to have favored a portfolio of OSS and CSS: the share of start-ups with 100% OSS-based business models never exceeds 16%. Thus, there is both, an institutional coexistence of mainly OSS-based and mainly CSS-based firms in the market as well as a coexistence of the both IPR-regimes within firms, here: within start-ups. The share of start-ups with hybrid strategies (i.e. business models based on a mix of OSS and CSS) increased over the years reaching a maximum of 68.42 % in the year 2008.

analyze the use of OSS vs. CSS. Accordingly, none of the firms founded in the years before 1984 reported to have used OSS at time of start-up. But of course this does not necessarily imply that all of them have entirely used "closed" code in the sense of CSS.

Since we could only raise information for start-ups that still existed until the time of our survey we have to be aware of a survivor bias in the data set. Examining the number of firms ordered by their reported year of founding illustrates this bias. Starting with the year 1984 the number of firms tends to increase reaching its peak in the year 2004. Comparing these numbers with general data about newly founded ICT businesses from the start-up panels of the Centre for European Economic Research (ZEW Mannheim)[7] shows that for earlier years the trend in our data (number of start-ups) does not coincide with the trend of the ZEW data. This is clearly driven by the fact that we have the survivor only. In addition, in our data is a drop in the number of start-ups of the year 2008, which does not coincide with the general trend of the ZEW data. This suggests an under representation of 2008 start-ups in our sample, which can be explained by a time-lag for being included in the heise data base, the source of addresses for our survey.

7.5 Results of Ordered Logit Analyses: Who Chooses OSS?

We want to explain the OSS-intensity in the different business fields by the individual characteristics of the respective start-up firms. As the dependend variable is of ordinal character—ranging from never OSS (value = 0) to always OSS (value = 4)—we applied ordered logit analyses (see Greene, 2008, for details). In order to avoid a survivor bias we estimate the models for the relatively recent cohorts of the years 2005 to 2008, but also for 2003 to 2006 (around the peak in 2004).[8] Since our sample of start-ups in the year 2008 may be biased due to a relatively incomplete coverage of that specific vintage, we also run all models for the 2005 to 2007. In addition, the analysis was also performe for all start-ups of the years 1984 to 2008 and 2003 to 2008.

As we perform ordered logit regressions we have to test for the parallel regression (proportional odds) assumption. Therefore we do the brant test

[7] We are indebted to the Centre of European Economic Research for providing these data.

[8] The two periods overlap by one year in order to have a sufficient number of observations available

for each of our models. Furthermore, we include two control variables for the business fields. In the tables represented in this chapter, we make use of two aggregated business field dummies: "further dev. software" and "new media & internet". The first dummy has a value of 1 if the founder has reported that software, further-developed by the start-up firm itself, was used in the business field.[9] The latter dummy is 1 if "web hosting", "web design and web service", or "services of new media agencies and related" was the start-up business field. We also run all ordered logit regression models with the complete set of detailed business field dummies (see Table E.5 the Appendix E.2 for the list of dummies). The results are very similar, but the brant test cannot be computed, as the disaggregated business field dummies cannot be retained in all binary logits.

In all groups of start-up cohorts—except the two most recent ones (period 2005 to 2008, Table 7.1, and 2005 to 2007, Table 7.2)—we find that the control variable for the age of a firm is significantly related to the use of OSS at the time of start-up, This means that older firms show a lower probability of having been an OSS-based start-up than more recently founded businesses. This may be interpreted in two ways. First, as our descriptive statistics indicate, OSS has been less common in earlier periods and has become more popular as a business concept in recent years. Second, in case that OSS start-ups should be to a higher degree subject to failure than CSS start-ups, this statistical relationship may be a result of a survivor-bias in the data. Because of this, in the following analysis we treat the results of the cohorts of 2003 to 2006 with care, and exclude those of 1984 to 2008 and 2003 to 2008. The descriptive statistics of the cohorts from 2003 to 2006, from 2005 to 2007 and from 2005 to 2008 can be found in the Appendix E.1, tables E.3, E.2 and E.1.

The results for the more recent cohorts (period 2005 to 2008, Table 7.1) that should not be strongly affected by a survivor-bias clearly indicate that OSS-based start-ups tend to be smaller in terms of personnel and in terms of capital invested. Accordingly, OSS start-ups are to a lesser degree constrained by the availability of capital: in the 2005 to 2008 cohorts they do

[9] This implies that either "selling own hardware with further-developed software", "selling third-party hardware with further-developed software", or "selling further-developed software" was a start-up business field. The term further-developed implies here that the software was further-developed by the respective start-up firm, not by a third-party.

not face special problems in convincing investors. In one model each, the founders of OSS firms have an on average higher qualification level and more experience than those who start a firm based on CSS. We cannot find any statistically significant evidence that OSS start-ups are more motivated by necessity than CSS-based new firms. The brant test indicates that only the models (7) and (8) are problematic.

The highly significant aggregated business field dummy "new media & internet" indicates that start ups from the "new economy" have a strong bias towards the usage of OSS, which is consistent with the descriptive statistics. This is also intuitively, as for example some of the most successful OSS projects are products used by the internet-economy, like e.g. the Lamp stack software for running web servers (see Section 1.2.3). Therefore, we also run the ordered logit regressions without the new media & internet group. The results remain quite robust: size in terms of staff and capital as well as lack of own capital are still significantly and negatively correlated with the OSS intensity.[10]

We also check whether there is a bias because of a "hardware effect", meaning that firms selling hardware have a significant different affection towards OSS than the software firms. As hardware firms may also systematically differ with respect to the tested firms properties, this could lead to distorted results. Therefore, we use as alternative business field control a hardware/software dummy in the regressions without the new media and internet start ups. However, this was never significant. Furthermore, the results do not change, if one concentrates on the software firms only, thus exclude the hardware-firms.

The recent cohorts without the year 2008, i.e. the period 2005 to 2007 (Table 7.2), show nearly the same results than the 2005 to 2008 period. Mainly, the results differ with respect to education: in the 2005 to 2007 cohorts the level of education of the founders has a positive impact on the OSS-intensity in seven out of nine models. The brant test for the 2005 to

[10]Without the new media and internet start-ups, there are only a few firms doing only OSS. Thus, for regressions that can be tested with the brant test, we must regroup the dependent variable in the sense that we merged the two highest categories ("mainly OSS" and "only OSS") in one. Furthermore we had to exclude the variable experience to run the brant test. The resulting models are approved by the brant test (except one model), but still lead to similar results as the original ones.

Table 7.1: OSS-Intensity of Start-Ups 2005 to 2008

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
size (staff)	-0.398**		-0.340*	-0.402**	-0.358*	-0.396**		-0.406**	-0.407**
	(0.028)		(0.070)	(0.027)	(0.062)	(0.032)		(0.029)	(0.026)
size (capital)	-0.406***	-0.443***	-0.401***	-0.407***	-0.407***	-0.386***	-0.422***	-0.433***	-0.403***
	(0.000)	(0.000)	(0.002)	(0.000)	(0.001)	(0.001)	(0.000)	(0.000)	(0.001)
lack of capital	-0.874**	-0.745*		-0.905**		-0.909**	-0.784*	-0.852**	-0.870**
	(0.041)	(0.071)		(0.019)		(0.039)	(0.064)	(0.040)	(0.041)
convince financiers	-0.0947	-0.228	-0.570			-0.287	-0.414	-0.0283	-0.0871
	(0.861)	(0.652)	(0.284)			(0.605)	(0.428)	(0.957)	(0.872)
edu	0.568	0.415	0.574	0.559	0.509	0.617*	0.463	0.507	0.557
	(0.117)	(0.241)	(0.121)	(0.122)	(0.167)	(0.091)	(0.192)	(0.138)	(0.121)
experience	0.854	0.833	0.911	0.878	1.113**			0.889	0.848
	(0.128)	(0.127)	(0.120)	(0.111)	(0.043)			(0.101)	(0.136)
necessity	-0.379	-0.389	-0.352	-0.370	-0.272	-0.425	-0.429		-0.366
	(0.333)	(0.294)	(0.364)	(0.340)	(0.469)	(0.282)	(0.250)		(0.352)
age of firm	0.0859	0.111	0.0836	0.0850	0.0717	0.0799	0.107	0.0736	
	(0.585)	(0.478)	(0.610)	(0.591)	(0.664)	(0.616)	(0.496)	(0.651)	
new media & internet	2.413***	2.429***	2.349***	2.419***	2.367***	2.284***	2.302***	2.367***	2.375***
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
further dev. software	1.500**	1.462**	1.517**	1.489**	1.434**	1.497**	1.461**	1.395**	1.480**
	(0.011)	(0.016)	(0.024)	(0.011)	(0.031)	(0.014)	(0.019)	(0.022)	(0.013)
(constants skipped)									
Observations	135	135	135	135	135	135	135	135	135
Pseudo R^2	0.169	0.159	0.157	0.169	0.154	0.163	0.153	0.166	0.169
df_m	10	9	9	9	8	9	8	9	9
probchi2	5.71e-09	1.50e-09	8.16e-10	1.99e-09	5.40e-11	2.08e-09	5.57e-10	2.25e-09	2.40e-09

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Table 7.2: OSS-Intensity of Start-Ups 2005 to 2007

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
size (staff)	-0.408**		-0.327*	-0.414**	-0.343*	-0.410**		-0.403**	-0.403**
	(0.036)		(0.089)	(0.035)	(0.081)	(0.042)		(0.040)	(0.028)
size (capital)	-0.435***	-0.462***	-0.426***	-0.438***	-0.437***	-0.432***	-0.458***	-0.451***	-0.433***
	(0.000)	(0.000)	(0.001)	(0.000)	(0.001)	(0.000)	(0.000)	(0.000)	(0.000)
lack of capital	-0.813*	-0.625		-0.844**		-0.822*	-0.636	-0.783*	-0.805*
	(0.070)	(0.134)		(0.040)		(0.071)	(0.134)	(0.071)	(0.067)
convince financiers	-0.0966	-0.265	-0.538			-0.175	-0.341	-0.0464	-0.107
	(0.865)	(0.611)	(0.334)			(0.760)	(0.516)	(0.932)	(0.849)
edu	0.715*	0.528	0.671*	0.712*	0.633*	0.767**	0.579	0.677*	0.707*
	(0.055)	(0.153)	(0.077)	(0.056)	(0.094)	(0.044)	(0.124)	(0.056)	(0.063)
experience	0.598	0.584	0.612	0.612	0.719			0.595	0.608
	(0.328)	(0.316)	(0.320)	(0.315)	(0.224)			(0.319)	(0.312)
necessity	-0.251	-0.214	-0.183	-0.242	-0.0861	-0.255	-0.213		-0.251
	(0.546)	(0.587)	(0.659)	(0.556)	(0.826)	(0.547)	(0.592)		(0.546)
age of firm	-0.0299	0.0706	0.0548	-0.0370	0.0231	-0.0661	0.0374	-0.0287	
	(0.907)	(0.777)	(0.828)	(0.884)	(0.927)	(0.792)	(0.879)	(0.911)	
new media & internet	2.334***	2.363***	2.273***	2.340***	2.291***	2.241***	2.272***	2.311***	2.341***
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
further dev. software	1.488**	1.386**	1.471*	1.479**	1.398*	1.529**	1.422**	1.406**	1.481**
	(0.030)	(0.049)	(0.059)	(0.029)	(0.068)	(0.026)	(0.044)	(0.043)	(0.028)
(constants skipped)									
Observations	123	123	123	123	123	123	123	123	123
Pseudo R^2	0.164	0.153	0.153	0.164	0.150	0.161	0.150	0.163	0.164
df_m	10	9	9	9	8	9	8	9	9
probchi2	1.98e-08	2.79e-08	6.65e-09	7.71e-09	9.58e-10	1.36e-08	1.65e-08	1.27e-08	7.90e-09

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Table 7.3: OSS-Intensity of Start-Ups 2003 to 2006

	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
size (staff)	-0.0705		-0.0465	-0.0236	-0.0217	-0.0705		-0.0530	-0.0346
	(0.565)		(0.691)	(0.834)	(0.847)	(0.565)		(0.662)	(0.767)
size (capital)	-0.215**	-0.227***	-0.242***	-0.233***	-0.245***	-0.215**	-0.227***	-0.204**	-0.230***
	(0.012)	(0.007)	(0.003)	(0.006)	(0.003)	(0.012)	(0.007)	(0.020)	(0.004)
lack of capital	-0.598*	-0.581*		-0.284		-0.598*	-0.581*	-0.621**	-0.476
	(0.057)	(0.061)		(0.304)		(0.056)	(0.059)	(0.044)	(0.141)
convince financiers	0.787**	0.752**	0.457			0.787**	0.752**	0.777**	0.770**
	(0.026)	(0.026)	(0.122)			(0.026)	(0.026)	(0.026)	(0.033)
edu	0.576**	0.556**	0.615**	0.624**	0.638**	0.576**	0.556**	0.593**	0.410
	(0.024)	(0.030)	(0.017)	(0.013)	(0.012)	(0.024)	(0.029)	(0.019)	(0.100)
experience	0.00242	0.00141	0.0583	-0.0401	0.00652			0.0405	-0.137
	(0.995)	(0.997)	(0.887)	(0.920)	(0.987)			(0.923)	(0.715)
necessity	0.301	0.285	0.335	0.292	0.315	0.301	0.285		0.496**
	(0.235)	(0.265)	(0.181)	(0.256)	(0.213)	(0.231)	(0.260)		(0.043)
age of firm	-0.365***	-0.358***	-0.325***	-0.357***	-0.334***	-0.365***	-0.357***	-0.404***	
	(0.006)	(0.006)	(0.009)	(0.005)	(0.006)	(0.005)	(0.006)	(0.001)	
new media & internet	1.277***	1.279***	1.243***	1.232***	1.224***	1.276***	1.279***	1.307***	1.258***
	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)	(0.000)
further dev. software	1.018**	1.037**	0.990**	1.116**	1.079**	1.018**	1.037**	1.071**	0.971**
	(0.024)	(0.022)	(0.031)	(0.024)	(0.026)	(0.023)	(0.021)	(0.016)	(0.036)
(constants skipped)									
Observations	260	260	260	260	260	260	260	260	260
Pseudo R^2	0.075	0.074	0.069	0.068	0.066	0.075	0.074	0.073	0.064
df_m	10	9	9	9	8	9	8	9	9
probchi2	1.26e-07	3.95e-08	3.75e-07	1.85e-07	3.10e-07	5.52e-08	1.68e-08	5.14e-07	2.39e-07

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

2007 cohorts can only be performed if the variable "convince financiers" is excluded. The results do not change much[11] and the brant test indicates that none of the regressions violate the proportional odds assumption

The cohorts containing the latest peak (2003 to 2006, Table 7.3) show quite similar results. However, the impact of staff vanish here and, as already mentioned above, the age of the firm becomes significant. As in the 2005 to 2007 period, education is significant throughout the models. It is quite remarkable, that different to the more recent cohorts, problems to convince investors seem to play a role as a special bottleneck for OSS-based start-ups in the models with the 2003 to 2006 cohorts. The respective measure is significant with a positive sign. However, as already mentioned, the results of the 2003 to 2006 period have to be treated with care, because of possible survivor bias.

With regard to our hypotheses we can conclude that the results of our analyses support hypothesis 1, stating that smaller start-ups—in terms of staff and capital—are more likely to be OSS-intensive than larger ones. According to hypothesis 2 we find that OSS-intensive start-ups report a lesser degree of capital shortages. This may be regarded an indication that OSS-based start-ups do not use OSS because they cannot afford CSS. In that sense OSS is in most cases not a strategy of the weak, but provides new entrepreneurial opportunities. A statistically significant relationship between problems of convincing potential financiers and OSS-intensity of the business concept (hypothesis 3) can only be found for the 2003-2006 cohorts. This suggests that the problems of convincing were driven by a lack of acceptance due to the relative newness of OSS business models in the older cohorts. In other words: if the problems of convincing are mainly caused by the fact that financiers assess OSS-based start-ups more risky, then "convincing financiers" would have to be significant also in the latest cohorts. If our indicators for experience and education are statistically significant they assume a positive sign so that hypothesis 4, stating that founders of OSS firms have lower levels of education and experience has to be rejected. According to our data, OSS seems to rather attract relatively highly qualified entrepreneurs than low quality ones. We find no evidence that OSS is particularly attractive for necessity-based start-ups (hypothesis 5).

[11] Only the significance niveau of the "lack of capital" variable increases.

7.6 Summary and Outlook

We find evidence that OSS-based business models enable firms to be smaller in terms of staff and capital so that they tend to experience capital shortages to a lesser degree than new businesses based on CSS. The possible disadvantage with regard to convincing potential financiers to invest into the firm vanishes over time: in the latest cohorts there is no longer any significant effect. Furthermore, the rising share of OSS-intensive start-ups over time indicates that this type of business concept is an attractive option for entrepreneurs. In this context it is important to notice that OSS does not represent a special attraction to relatively low-qualified entrepreneurs or to entrepreneurs who start a firm primarily out of necessity. We even find some evidence that founders of OSS-based businesses have a relatively high level of both, experience and education.

This results show a clear effect of institutions, here: the IPR regime, on the characteristics of market entry. This study therefore contributes to research on IPR and entrepreneurship (e.g. Simcoe et al., 2009; Libecap, 2004). However, the effect of the different IPR regimes on the characteristics of the ICT start-ups is of a rather complex nature. Obviously, simple hypotheses based on the lower individual resources that are needed for an OSS entry tell, at best, only a small part of the story. Further work should try to study these effects in more detail. It would be particularly interesting to learn more about the business models of OSS start-ups and how they differ from those of CSS firms. Since many start-ups are based on elements of OSS and of CSS it would be interesting to know more about the dynamics of this co-existence at the firm level. Do firms that start with a high share of OSS over time switch to more CSS? Do mainly CSS-based firms tend to increase their share of OSS? What makes these types of business models successful? How does the emergence of OSS firms affect the innovative performance of markets? Does the voluntary division of labor in the OSS community drive innovation processes? If yes, what kind of innovation, mostly incremental innovation, mostly radical innovation, or both? These are important questions for further research in the field.

8 Summary of the Dissertation and Outlook

This dissertation empirically and theoretically analyzes two coexisting IPR regimes, namely OSS and CSS. Individual institutional choices, the actions under the particularIPR regime (including the strategic interactions) and the resulting payoffs lead to the fact that OSS and CSS coexist. This chapter summarizes the findings of the dissertation and then discusses the possibility to derive policy implications. The chapter finishes with a section about further research.

8.1 Summary of Results

While OSS development is a global phenomenon, most (active) developers come from developed countries. This is the result of Chapter 2, which shows the geographic origin of registered developers at SourceForge in 2006. Information about the developers' IP address, email address, indicated time-zone, and the number of posted messages deliver detailed country data about the number of registered developers, active developers and their activity level. This unique dataset shows that 85% of all active developers come from OECD countries. Nevertheless, differences in the supply side of OSS per country cannot solely be explained by GDP and access to the internet. To explain these differences further, country-specific aspects like culture and institutions have to be taken into account.

The cross-country study presented in Chapter 3 shows that individual decisions for OSS are strongly influenced by country-specific institutional and cultural factors. Social capital interpreted as interpersonal trust has a positive impact on the number of OSS developers as well as on the OSS activity level. Furthermore, a culture characterized by individualism/self-determination and an optimistic view of scientific progress favors OSS ac-

tivities. And finally, a low degree of regulation as well as good protection of IPRs supports OSS. This study contributes to the understanding of the role of cultural and institutional factors in general as well as in particular with respect to OSS. Additionally, it improves the understanding of the supply-side of OSS in two ways. First, the findings confirm some results from microeconomic research on OSS. The regression results support the view that OSS has similarities to technical science and scientific culture. Also in line with the findings is the following view, summing up some common research results: OSS development is a public good game where interpersonal trust (as the basis for reciprocal behavior etc.) helps to play the 'right' equilibrium, while the individual motives of the players are of extrinsic as well as intrinsic nature. Second, the results of the study help to clarify some aspects. OSS is indeed rooted in the Western capitalist culture. OSS is based on an entrepreneurial spirit which is linked to individualism, and the results regarding regulation point to the commercial aspect of OSS. Furthermore the positive impact of IPR protection supports the view that OSS is not anti-IPR but a new IPR paradigm. OS is a new ownership concept for digital goods, with the OSS licenses define the intellectual property of the code.

OSS is thus a new intellectual property paradigm, which coexists with the CSS paradigm. In the tradition of the property rights theory, Chapter 4 asks "Which problem of internalization leads to the dichotomy of OSS and CSS?" The analysis shows that the first-best allocation of IPRs of a source code cannot be achieved. OSS and CSS are two pragmatic second-best solutions. The reason is a combination of ex-post transaction costs with the characteristics of software. Some applications imply access to the source code. Here the rights can de facto not be separated and must stay bundled. A code owner has therefore to decide whether to exclusively claim the whole bundle or not. Because of non-rivalry in use and feedback-effects, the source code is a non- and anti-scarce resource. This has implications for the rationale to claim all rights. Forgoing rights can enable feedback effects that yield a higher payoff than the rights themselves. Furthermore, it is not costly to forgo rights that cannot be traded anyway because of transaction costs. Therefore, code owners decide whether to exclusively claim the whole bundle or not, based on individual cost-benefit analysis. This is technically mirrored by the question whether the source code is 'closed' or

'open'. Therefore CSS is developed within firms, based on exclusive ownership of the code and only the separable rights are traded. While CSS is developed only by the programmers hired by the firm, the principle of OSS aims to attract a wider set of participants. OSS minimizes exclusive ownership and makes use of passive control rights. The OSS licenses are non-specific contracts that permit the transfer of the whole set of rights. From a social welfare perspective OSS and CSS both offers specific benefits and drawbacks, which are mirror inverted. The advantage of CSS is the benefit of exclusive ownership: direct control, internalization of external effects, etc. The disadvantage of CSS results from the boundaries of a firm: transaction costs inhibit contracting with all possible contributors, not all possible feedback effects are generated, thus: the source code is not optimally used. Because of its openness, OSS can create more spillover and feedback-effects, but lacks direct control. Therefore the coexistence of OSS and CSS is probably welfare superior to pure OSS or CSS scenarios.

Chapter 5 and 6 focus on the coexistence of OSS and CSS in terms of the chosen business strategy of firms. The two chapters analyze this topic in a general two-stage model that goes beyond the existing literature. It is not restricted to the software industry but covers OSS vs. CSS business models from the whole ICT industry. Furthermore, the model-setup allows one to vary the contributions that come from the non-commercial community. This allows for analysis of situations without non-paid volunteers. The model analyzes the 'economics of commercial OS' in a general way, and can be readily applied to other examples of OS (e.g. OS biotechnology). Thus, 5 and 6 present the economics of OS vs. CS business models, and analyze welfare implications.

Chapter 5 concentrates on the strategic nature of OS vs. CS business models and the role of OS license type. The model combines aspects of non-cooperative R&D with the theory of differentiated oligopolies. In stage one, firms develop software, either as OS or CS. If the license is of liberal type (e.g. the BSD), firms can use an OS-CS-mix, otherwise not. In stage two, firms bundle this software with complementary products and compete à la Cournot. The software determines the quality of the products. Thus OS lets firms avoid quality competition as they can cooperate on quality without an explicit contract. The model allows for horizontal product differentiation in stage two. It turns out that CS-decisions are always strategic

substitutes, while OS-decisions can be strategic complements. Furthermore, CS is a strategic substitute for OS and vice versa. The type of OS license plays a crucial role. In the case of liberal licenses, firms only develop OS code if the products are horizontally differentiated such that quality competition is low (e.g. mobile phones vs. servers etc.). Nash-equilibria with firms producing OS code for all parameters only exist for restricted licenses (e.g. the GPL). In the next step, the equilibrium ratios of OS/CS firms in a mixed industry with restricted licenses are analyzed. Given the ratio of OS/CS firms needed to stabilize the market against further entry OS, firms offer lower quality than their CS-rivals. Where horizontal product differentiation is low (e.g. mobile phones vs. mobile phones), CS-based products have the largest market share.

Chapter 6 concentrates on welfare aspects and mainly focuses on cases without non-paid OSS volunteers (all OS code is commercial OS). The chapter first deepens the analysis of OS and CS code output, as this is the basis of understanding the welfare effects. From a social point of view, the cost-saving benefits from OS code-sharing are contrasted by the OS cartel effect: code-sharing guarantees that no OS firm can offer better software than any other OS firm. This suppresses quality competition between OS firms and restricts their code output. Competition from CS firms weakens this quality-cartel effect. As result, the equilibria of mixed industries offer higher welfare than Pure-OS or Pure-CS. Furthermore, Pure-OS (Pure-CS) industries are sometimes stable against CS (OS) entry so that the mixed OS/CS state never occurs. Even where mixed OS/CS industries do exist, the proportion of OS firms needed to stabilize the market against entry is always larger than the target ratio required to optimize welfare. Chapter 6 then discusses various government interventions for addressing this imbalance with tax policy, funding of OS development, and procurement preferences. It turns out that the first-best solution in the model is to tax OS firms and grant tax breaks to CS firms. Conversely, government interventions that fund OS development or establish procurement preferences for OS software increase the gap between desired and actual OS/CS ratios still further. Despite this, funding OS development can still improve welfare by boosting total (private plus government) OS investment above the levels that a private OS cartel would deliver.

For the sake of simplicity, the firms in the model of Chapters 5 and 6 are symmetric, except for the horizontal and vertical (quality) differences of their products. But different individual resources are one reason why some economic agents choose the OS regime and others the CS regime (see Chapter 4). This should also be true for e.g. new businesses based on OSS vs. CSS. Furthermore, the fact that there can be an inferior lock-in in Pure-OS and Pure-CS industries points to the importance of entry. Therefore Chapter 7 of this dissertation focuses on entry, namely on OSS- vs. CSS-intensive start-ups in the German ICT sector. The empirical study of Chapter 7 analyzes the characteristics of new businesses in the German ICT industry distinguished by their choice between the two IPR-regimes, OSS and CSS. The analysis is based on a survey among founders of ICT firms in Germany conducted in fall 2009. This is the first study that directly compares OSS- and CSS-based start-ups and analyzes which aspects shape their institutional choices. The results contribute to research on OSS and CSS and to the understanding of the relationship between institutions and entrepreneurship. The findings are that firms with OSS-based business models tend to be smaller in terms of staff and capital. OSS-firms also experience less shortages of capital. Furthermore, the data show that OSS business models seem to be established nowadays, as only OSS-intensive start-ups in older cohorts have larger problems than their CSS counterparts to convince potential financiers to invest. The data do not indicate that the lower entry barriers for OSS firms are particularly attractive for start-ups with low human capital endowment or to necessity-motivated entrepreneurs.

8.2 Policy Implications

This section briefly discusses the possibility of deriving policy implications from the results. Note that the main purpose of the research presented in this dissertation is not to derive concrete, i.e. specific and detailed policy implications. Such policy advice would need further research. Nevertheless it is possible to derive some general statements.

The coexistence of OSS and CSS can be explained with the fact that both are second best solutions. Both IPR regimes have their specific advantages

and disadvantages, and they are complements rather than substitutes. Additionally, the coexistence itself leads to effects that improve the situation, like the interaction between OS and CS firms: the OS cartel effect is reduced by competition from CS-rivals. From this one can derive the rule that policymakers should not favor OSS over CSS or vice versa. In general, neither the OS nor the CS paradigm should exist alone. The theoretical results indicate that a situation where OS and CS principles coexist lead to higher welfare. Therefore policy should favor the coexistence.

Furthermore, the analysis of different government interventions in Chapter 6 shows that direct support of OS firms reduces welfare. This is true for tax breaks, but also for procurement preferences. Here the theoretical analysis provides an argument against the pro-OSS procurement policy of many governments. However, the model suggests that government can improve welfare by providing OSS. In practice this would imply that government funds certain OSS projects. Clearly, such a policy would yield several problems which are typical for subsidies. Government has to decide how much money goes to which projects. First, policymakers do not know which projects deliver the most social benefit (pretence of knowledge, von Hayek, 1975). Second, this would induce rent-seeking activities. The members of the different OSS projects (including the firms involved) have high incentives to lobby in order to get financial support for their project.

According to the model, there can be an OS-only or CS-only industry lock-in. Policy should be aware of this fact, although this does not necessarily imply the need for government interventions. Rather, persisting situations without the coexistence of the two IPR regimes should lead to further economic analysis. The reader should moreover note that lock-ins exist only for a small set of parameters in the model. Entrants are often entrepreneurs in the sense that they create new products or modify existing ones. This would change the parameters of the model (i.e. the degree of horizontal product differentiation).

Ghosh (2006)—a study for the European Commission—supports pro-OSS government policies. Ghosh (2006) states that OSS and OSS-based business models can foster innovation and the competitiveness of the ICT sector, as OSS can encourage the creation of small and medium enterprises and jobs. The reader should note that Ghosh (2006) is not based on an empirical comparative analysis of OSS and CSS business models. The study of

Chapter 7 provides a comparative analysis of start-ups in the German ICT sector. The results seem to indicate that OSS indeed fosters new business foundations. OSS enables to start-up with less capital and staff. But well-founded policy implications cannot be drawn from these results unless the long-term impacts are taken into account. This fact points to the need for further research.

8.3 Further Research Questions

The impact of OSS- vs. CSS-based business models on the long-term performance of firms is still not explored. Here the dataset used in Chapter 7 contains further information. This should therefore be used to analyze the impact of the chosen IPR-Regime on the performance measures "growth of staff". The research question therefore is: What is the link between OSS-intensity and firm growth (in terms of employees)? This would refer to realized growth as well as expected growth, the latter based on the expected staff in three years. Furthermore it is of interest how the OSS-intensity within the (faster-growing) firms evolves. Over time, the share of OSS-intensive start-ups increased. But does the OSS-intensity of firms also increase over time (e.g. a general trend towards OSS), or do firms who have entered as OSS-intensive start-ups switch to CSS-based business models?

The model of Chapters 5 and 6 shows that restricted OS licenses ensure commercial OS, and CS licensing ensures CS development by firms. This result is based on a distinction between CS license and restricted vs. liberal OS license. In reality, firms and project leaders use a scope of licenses. It could be of interest to further differentiate the types of licenses in the model. How restrictive must a license be to ensure private investments? In addition, it is known that firms contribute back for reasons that are beyond the scope of the model. This is especially important if the license is liberal. For example firms contribute back because of ideological reasons, because of learning effects, or because of reciprocal behavior of the community (they trade code for help), etc. see section 1.3.4. An analysis of firm behavior and the underlying motives is provided by Rossi and Bonaccorsi

(2006). Although these motives are well explored, there is still a lack of a simple model that captures these incentives.

The model of Chapters 5 and 6 also shows that with restricted licenses, commercial OS development can establish even in the absence of non-commercial OS contributors. This is an important result, especially for research on OS beyond software. Research on OS biotechnology still lacks theoretical work. Model extensions and specifications could be used to analyze OS and CS principles regarding the several stages of the drug discovery pipeline.

Based on the property rights theory, the framework of Chapter 4 provides the rationale and conditions for the coexistence of OS and CS principle for digital goods. First, a feedback mechanism (anti-scarce use) which implies access is combined with an access problem that yields non-separable rights. Second, transaction costs lead to the fact that for some actors it is a rational choice to forgo the bundled rights. Applying this general framework to the case of OS biotechnology can be fruitful. This might show some limits of the developed framework and can help to further develop it. Other possible examples for the coexistence of OS and CS principles should also be taken into account. Based on this, the notation of Chapter 4 can be used to develop a formal theory of (intellectual) property rights of scarce, non-scarce and anti-scarce goods, based on set theory.

The impact of culture and formal and informal macro-level institutions on OSS activities is analyzed in Chapter 3. On the micro-level, OSS development is characterized by an interplay of formal and informal institutions. Legally, the OSS licenses are based on copyright law. But from a cultural point of view, they are based on the so-called 'hacker ethics'. The OSS-project hierarchies are based on formal institutions like licenses or trademarks and on informal institutions like community rules. It would be interesting to know whether and how the macro-level (country-specific cultural and institutional factors) shapes the micro-level institutions of OSS: Do the governance structures of projects with participants mainly from one country or region differ from more global ones? Is there an 'Asian' vs. a 'European' type of OSS governance?

Finally, the history of CSS and OSS gives some insight into the micro-level of endogenous institutional change: institutional entrepreneurs can be driven by profits, efficiency and other 'rational' motives, but also by non-

commercial intentions including ideology. Created for whatever reason, the institutional inventions then have to survive the competition with alternative (new or already established) institutions and arrangements. The history of OSS demonstrates how conscious invention can lead to unintended consequences (Did Richard Stallman intend to lay ground for today's OSS-based business models?), a spontaneous order in the sense of von Hayek (1967). Formal theory about institutional change should take this into account. It could be of interest to incorporate 'unintended consequences' into models of institutional change. In other words: While some institutional changes are induced by intention, some are induced by mutation. A theory of institutional change should take both aspects into account. In addition to this, the history of CSS and OSS also points to an 'action vs. reaction' relationship: some actors introduced the CSS-principle (action) which lead to the institutionalization of OSS by e.g. the GPL (reaction). Finally, both types of IPR arrangements, OSS and CSS, do coexist until today, i.e. the institutional competition has not yet led to a dominating role of one of these alternative regimes over the other. If further examples of coexisting institutional arrangements exist, a theory of 'institutional competition' could be developed.

A Appendix to Chapter 3

Table A.1: OSS Developers per 1,000 Inhabitants (without SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-0.777	-0.715	0.843	0.721	-0.801	-0.755
	(0.602)	(0.627)	(0.428)	(0.520)	(0.602)	(0.618)
education	0.232	0.235	0.317	0.253	0.0937	0.0969
	(0.583)	(0.579)	(0.374)	(0.429)	(0.817)	(0.811)
inet users	1.419***	1.415***	1.070*	1.063*	1.581***	1.576***
	(0.008)	(0.009)	(0.055)	(0.050)	(0.003)	(0.003)
prefer new ideas	2.047	1.722			1.053	0.822
	(0.473)	(0.519)			(0.733)	(0.779)
science advance help	1.104*	1.038*			1.022*	0.974
	(0.056)	(0.080)			(0.078)	(0.104)
self-determ/indiv	1.125*	1.084*	0.476	0.564	1.185**	1.154**
	(0.056)	(0.055)	(0.383)	(0.328)	(0.049)	(0.046)
intpersonal trust	1.183**	1.196***	0.587*	0.542*	1.177**	1.187**
	(0.010)	(0.010)	(0.073)	(0.099)	(0.020)	(0.018)
competition/merit	-0.133		0.264		-0.0976	
	(0.671)		(0.380)		(0.735)	
IPR protection	0.317	0.321	0.309	0.299	0.451	0.453
	(0.341)	(0.316)	(0.357)	(0.374)	(0.166)	(0.155)
degree of regulation	-1.016	-1.004	-0.835	-0.816		
	(0.126)	(0.114)	(0.105)	(0.122)		
_cons	-0.929**	-0.903*	-0.300	-0.218	-1.095**	-1.075**
	(0.042)	(0.058)	(0.350)	(0.459)	(0.020)	(0.028)
N	60	60	70	70	60	60
adj. R^2	0.786	0.790	0.759	0.760	0.777	0.781
d. of freedom	49	50	61	62	50	51

p-values in parentheses

* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Table A.2: Active OSS Developers per 100 Inhabitants (without SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-1.964	-1.922	2.596	2.263	-2.019	-2.016
	(0.525)	(0.530)	(0.315)	(0.411)	(0.530)	(0.528)
education	0.578	0.580	0.563	0.388	0.254	0.254
	(0.533)	(0.529)	(0.475)	(0.578)	(0.775)	(0.772)
inet users	2.519**	2.516**	1.900	1.881*	2.898**	2.898***
	(0.027)	(0.026)	(0.108)	(0.099)	(0.010)	(0.010)
prefer new ideas	4.948	4.728			2.618	2.602
	(0.410)	(0.405)			(0.684)	(0.674)
science advance help	1.909*	1.864*			1.718	1.714
	(0.082)	(0.096)			(0.134)	(0.143)
self-determ/indiv	2.641**	2.613**	1.124	1.366	2.780**	2.778**
	(0.038)	(0.034)	(0.354)	(0.291)	(0.032)	(0.028)
intpersonal trust	2.668***	2.677***	1.144*	1.020	2.655**	2.656**
	(0.005)	(0.005)	(0.087)	(0.140)	(0.014)	(0.013)
competition/merit	-0.0900		0.721		-0.00699	
	(0.897)		(0.274)		(0.991)	
IPR protection	1.350*	1.353*	1.037	1.011	1.666**	1.666**
	(0.073)	(0.065)	(0.183)	(0.200)	(0.027)	(0.026)
degree of regulation	-2.382*	-2.374*	-1.981**	-1.928*		
	(0.065)	(0.056)	(0.043)	(0.061)		
_cons	-1.961*	-1.943*	-0.621	-0.397	-2.350**	-2.349**
	(0.052)	(0.064)	(0.365)	(0.515)	(0.026)	(0.030)
N	60	60	70	70	60	60
adj. R^2	0.809	0.813	0.781	0.780	0.797	0.801
d. of freedom	49	50	61	62	50	51

p-values in parentheses
* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

Table A.3: OSS Activity Level: Messages per 10,000 Inhabitants (without SLD)

	Ia	Ib	IIa	IIb	IIIa	IIIb
GDP	-1.943	-1.885	2.445	2.209	-1.978	-1.944
	(0.288)	(0.300)	(0.273)	(0.344)	(0.285)	(0.293)
education	0.590	0.592	0.472	0.348	0.385	0.387
	(0.465)	(0.461)	(0.492)	(0.576)	(0.626)	(0.620)
inet users	1.630**	1.626**	1.220	1.207	1.869**	1.866**
	(0.048)	(0.049)	(0.140)	(0.129)	(0.024)	(0.024)
prefer new ideas	4.012	3.711			2.539	2.372
	(0.380)	(0.389)			(0.593)	(0.603)
science advance help	1.148*	1.087			1.028	0.993
	(0.094)	(0.106)			(0.148)	(0.151)
self-determ/indiv	2.741***	2.702***	1.498	1.668*	2.829***	2.806***
	(0.006)	(0.005)	(0.103)	(0.089)	(0.006)	(0.005)
intpersonal trust	2.676***	2.688***	1.305**	1.218**	2.667***	2.674***
	(0.000)	(0.000)	(0.013)	(0.027)	(0.001)	(0.001)
competition/merit	-0.123		0.511		-0.0708	
	(0.851)		(0.329)		(0.907)	
IPR protection	0.980*	0.984*	0.492	0.473	1.179**	1.181**
	(0.100)	(0.087)	(0.439)	(0.468)	(0.043)	(0.040)
degree of regulation	-1.505	-1.494*	-1.218*	-1.181		
	(0.103)	(0.085)	(0.071)	(0.105)		
_cons	-1.546*	-1.522*	-0.527	-0.369	-1.792**	-1.777*
	(0.079)	(0.091)	(0.366)	(0.482)	(0.048)	(0.052)
N	60	60	70	70	60	60
adj. R^2	0.835	0.838	0.800	0.799	0.829	0.832
d. of freedom	49	50	61	62	50	51

p-values in parentheses

* $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$

B Appendix to Chapter 4

This Appendix presents a more detailed notation of IPRs. As Chapter 4 is about software we thus refer to IPRs in the following. However, in principle the notation can be applied to PRs and IPRs.

Following Furubotn and Richter (2005); Eggertsson (1990); Hart and Moore (1990), and others, we distinguish coordination rights (usus and abusus) from residual rights (usus fructus and alienation rights). The complete set of rights is defined as $H = \{H^c \cup H^r\}$, with H^c as the set of coordination rights, and H^r as the set of residual rights. Let $h \in H$ denote one property right.

At first, coordination rights are defined: The conjunction of a coordination right $h^c \in H^c$ with a resource X (we write "$h^c : X$") leads to a distinction-criteria between the applications that are covered by the IPR and those which are not. Notice that there is no need to know the whole set of possible applications, as the distinction-criteria yields a selecting-rule r that tells whether a $y \in Y(X)$ is covered by the IPR or not: $r : y \rightarrow [0,1] \ \forall\, y \in Y$. This leads to

$$h^c : X \longrightarrow \{y \in Y \mid r(y) = 1\}. \tag{B.1}$$

For example, let $\mathbf{h}_u$ denote the vector of all usus rights, $\mathbf{h}_a$ the vector of all abusus rights, and $\mathbf{h}_{u\&a}$ all usus and abusus rights. This yields e.g.

$$\mathbf{h}_u : X \longrightarrow \{y \in Y \mid y = f(Z)\} = Y(Z), \tag{B.2}$$

$$\mathbf{h}_{u\&a} : X \longrightarrow \{y \in Y \mid y = f(V)\} = Y(V). \tag{B.3}$$

Next is to define the residual rights, where whave to distinguish between usus fructus and alienation rights: Let $\mathbf{h}_f$ denote the vector of all usus fructus rights, and π the payoff gained from y, then

$$\mathbf{h}_f : X \longrightarrow \{\pi \mid \pi = f(y),\ y \in Y\}. \tag{B.4}$$

The right to transfer IPRs of a resource has to be represented in a slightly different way, as it is a 'right on rights'. Let h^h denote the alienation right regarding h. An individual holding an alienation right with respect to h can therefore *decide* wether to keep this right, or transfer it, i.e. do not hold it anymore:

$$h^h = \text{'decision'} \circ h = \begin{cases} 0 & \text{if right } h \text{ is transferred,} \\ h & \text{if right } h \text{ is not transferred.} \end{cases} \tag{B.5}$$

Let $\mathbf{h}^h$ be the vector of all alienation rights. Holding alienation rights on a resource X is then formally given by

$$\mathbf{h}^h : X = \begin{pmatrix} h^{h^1} : X \\ \vdots \\ h^{h^n} : X \end{pmatrix} = \begin{pmatrix} [0,1] \cdot h^1 : X \\ \vdots \\ [0,1] \cdot h^n : X \end{pmatrix}. \tag{B.6}$$

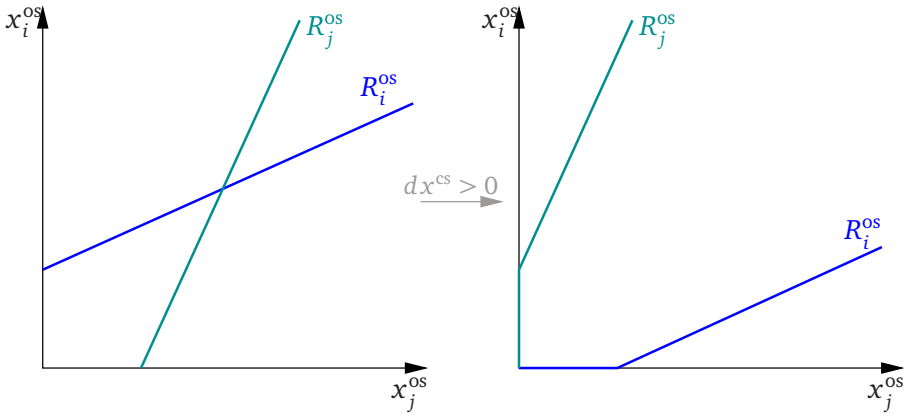
Notice, that there exist "recursive" alienation rights. A recursive alienation right is the right to transfer an alienation right. This means, that

$$\exists\, h^k \in \{h^1 \dots h^n\} \text{ s.t. } \left[h^k = h^{h^j} \text{ with } h^j \in \{h^1 \dots h^n\},\, h^j \notin \mathbf{h}^h \right]. \tag{B.7}$$

C Appendix to Chapter 5

If $\phi > \phi^{os}_{\bar{x}}$, then there is always only one unique equilibrium in OS development, and furthermore CS output decreases x^{os*}. Figure C.1 depicts this

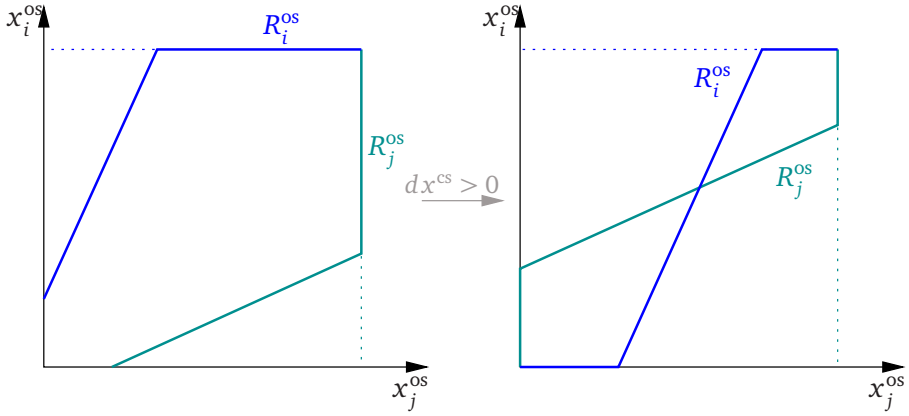
Figure C.1: OS-Equilibrium and the Impact of x^{cs} if $\phi > \phi^{os}_{\bar{x}}$



with an example of $z = 2$ OS-firms: if x^{cs} is moderate, OS firms develop $x^{os} > 0$ (left hand), while if there is much x^{cs}, x^{os} production is driven to zero (right hand).

For $\phi < \phi^{os}_{\bar{x}}$, the situation is more complex. In this case the slope of the OS-reaction functions (with respect to the other firms' OS) are greater than one. This implies that multible equilibria can exist. Again, we illustrate this graphically for the case of $z = 2$. If $x^{cs} = 0$ or low enough, then the OS firms together develop the cutoff, see the left hand of figure C.2. But if x^{cs} is high enough, then the OS-rection functions are shifted so that multible equilibria exist. The right hand side of figure C.2 depicts such a situation. Without additional assumptions, each of the three equilibria are

Figure C.2: OS-Equilibrium and the Impact of x^{cs} if $\phi < \phi^{os}_{\bar{x}}$



plausible. Hence, either the OS firms develop the cutoff, or the equilibrium $0 < X^{os} < \bar{x}$ establishes, or the OS firms develop no code at all.

D Appendix to Chapter 6

D.1 OS versus a 'Real' OS-Cartel

In this section we compare the outcome of individual investment decisions by OS firms (the normal case analyzed in our model) against a hypothetical formal cartel in which firms agree to coordinate their activities to maximize joint profits. We find that individual decisions produce less output than a 'real' OS cartel would supply. Because of this underprovision/public good problem, the suppression of quality competition among OS firms only leads to a 'second best' cartel compared to the outcome under a 'real' quality cartel.

D.1.1 OS-Firms, No Formal Cartel

In case of OS with competition, each OS firm $i = 1 \dots n$ maximizes its profit by choosing the optimal x_i^{os}:

$$max_{x_i^{os}} \pi_i = \frac{\left(1 + x_i^{os} + \sum_{j \neq i} x_j^{os}\right)^2}{h^2} - \frac{1}{2}\phi x_i^{os}\left(x_i^{os} + \sum_{j \neq i} x_j^{os}\right)$$

The first derivative

$$\frac{\partial \pi_i}{\partial x_i^{os}} = \frac{2\left(1 + x_i^{os} + \sum_{j \neq i} x_j^{os}\right)}{h^2} - \frac{1}{2}\phi\left(x_i^{os} + \sum_{j \neq i} x_j^{os}\right) - \frac{1}{2}\phi x_i^{os} \stackrel{!}{=} 0$$

then yields optimal decision of each firm, i.e. firm i's reactionfunction

$$x_i^{os} = f(\sum_{j \neq i} x_j^{os}) = \frac{1 - \left(\frac{1}{4}\phi h^2 - 1\right) X_{-i}^{os}}{\frac{1}{2}\phi h^2 - 1}, \ \forall \phi > \frac{2}{h^2}$$

By symmetry this same reaction function governs all $i = 1 \dots n$ firms. This determines the equilibrium value of x^{os}, and hence the equilibrium value of $X^{os} = \sum_i x_i^{os} = n \cdot x^{os}$ given by

$$X^{os,*} = n \cdot x^{os,*} = \frac{n}{\frac{1}{4}\phi h^2 (1+n) - n}$$

D.1.2 A Formal OS Quality-Cartel

Consider a 'real' cartel in which firms develop code $X^{os} = \sum_i x_i^{os}$ so as to maximize joint profits $\Pi = \sum_i \pi_i^{os}$. Then profit maximization requires:

$$\max_{X^{os}} \Pi = \sum_i \pi_i^{os} = n\frac{(1+X^{os})^2}{h^2} - \frac{1}{2}\phi X^{os2}$$

The first derivative

$$\frac{\partial \Pi}{\partial X^{os}} = n\frac{2(1+X^{os})}{h^2} - \phi X^{os} \stackrel{!}{=} 0$$

then yields the cartel-output given by

$$X^{os,\,cartel} = \frac{n}{\frac{1}{2}\phi\, h^2 - n} \; \forall \phi > \frac{2n}{h^2}$$

D.1.3 Comparing Outcomes

For all $\phi > \frac{2n}{h^2}$ (the second order condition of the cartel), the cartel produces more code than individual firms do:[1]

$$X^{os,\,cartel} > X^{os*}.$$

Furthermore firms also earn higher profits, i.e.

$$\pi_i^{os,\,cartel} > \pi_i^{os*}.$$

Finally, in terms of welfare we have

$$W^{os,\,cartel} < W^{os*}.$$

[1] If the second-order condition for cartels is not satisfied, i.e. $\phi < \frac{2n}{h^2}$, the cartel produces software up to the cut-off.

D.2 Welfare

Our welfare analysis is based on Hsu & Wang (2005) who provide a general welfare analysis for differentiated oligopolies. Applied to our model, this yields the consumer surplus

$$A = \frac{1}{2}\left((1-\gamma)\left(zq_{i\in Z}^2 + rq_{i\in R}^2\right) + \gamma\left(zq_{i\in Z} + rq_{i\in R}\right)^2\right)$$

with

$$q_{i\in Z} = \frac{(1 + zx^{\text{os}} - r\theta\,(x^{\text{cs}} - zx^{\text{os}}))}{h}$$

and

$$q_{i\in R} = \frac{(1 + x^{\text{cs}} - z\theta\,(zx^{\text{os}} - x^{\text{cs}}))}{h}.$$

The producer surplus is given by

$$B = z \cdot \pi_{i\in Z} + r \cdot \pi_{i\in R},$$

with profits given by

$$\pi_{i\in Z} = \frac{(1 + zx^{\text{os}} - r\theta\,(x^{\text{cs}} - zx^{\text{os}}))^2}{h^2} - \frac{1}{2}\phi z x^{\text{os}2}$$

and

$$\pi_{i\in R} = \frac{(1 + x^{\text{cs}} - z\theta\,(zx^{\text{os}} - x^{\text{cs}}))^2}{h^2} - \frac{1}{2}\phi x^{\text{cs}2}.$$

D.2.1 Pure Cases

In case of Pure-OS ($n = z$) we obtain

$$X^{\text{os}} = n \cdot x^{\text{os}} = \frac{n4}{\left(\phi h^2\,(1+n) - 4n\right)},$$

and thus welfare is given by

$$W_n^{\text{os}} = A_{n=z} + B_{n=z} = (1+h)\,\frac{n}{2}\frac{(1+nx^{\text{os}})^2}{h^2} - \frac{1}{2}\phi\,(nx^{\text{os}})^2\,.$$

In case of Pure-CS ($n = r$) we obtain

$$x^{cs} = \frac{2(1+(n-1)\theta)}{\left(h^2\phi - 2(1+(n-1)\theta)\right)},$$

and thus welfare is given by

$$W_n^{cs} = A_{n=r} + B_{n=r} = (1+h)\frac{n}{2}\frac{(1+x^{cs})^2}{h^2} - n\frac{1}{2}\phi x^{cs2}.$$

We can now calculate the *difference in welfare* ($\mathcal{W}$) for a given n:

$$\mathcal{W}_n = W_n^{os} - W_n^{cs} = \frac{n}{2}(1+h)\frac{(1+x^{os}n)^2 - (1+x^{cs})^2}{h^2} - \frac{n}{2}\phi\left(nx^{os2} - x^{cs2}\right).$$

D.2.2 Mixed Cases

In case of a mixed industry we obtain

$$X^{os} = z \cdot x^{os} = z\frac{(1+\theta r)\left(\theta r(1+(n-1)\theta) - \psi\right)\beta}{(1+\theta r)\theta^2 r(1+(n-1)\theta)z^2 - \psi\chi}$$

and

$$x^{cs} = \frac{(1+(n-1)\theta)\left(z^2\theta(1+\theta r) - \chi\right)\beta}{(1+\theta r)\theta^2 r(1+(n-1)\theta)z^2 - \psi\chi}$$

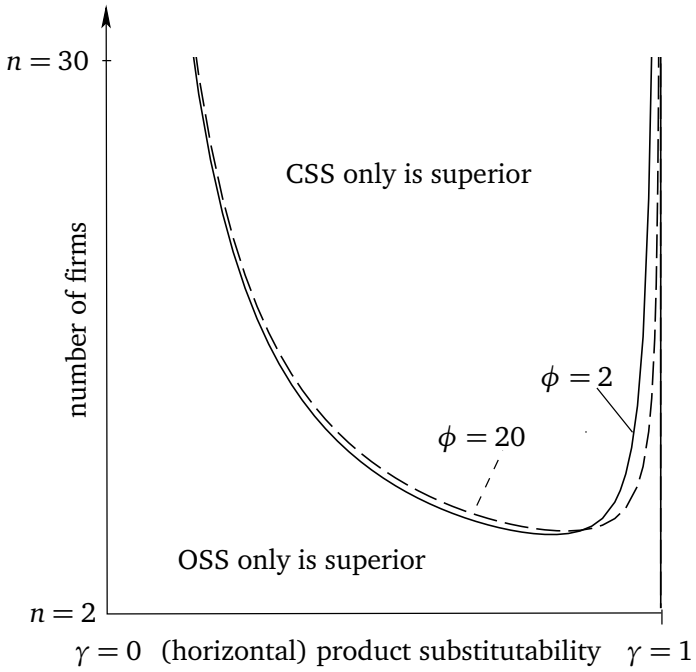
Inserting these in our quantity and profits functions (see above) lets us numerically calculate total welfare according to the expression

$$W = \frac{1}{2}\left((1-\gamma)\left(zq_{i\in Z}^2 + rq_{i\in R}^2\right) + \gamma\left(zq_{i\in Z} + rq_{i\in R}\right)^2\right) + z\cdot\pi_{i\in Z} + r\cdot\pi_{i\in R}.$$

D.3 Different ϕ and Welfare Comparison of Pure Cases

Here we show that adopting values of ϕ different from 2 does not substantially change our welfare comparison of Pure-OS and Pure-CS industries. Figure D.1 shows how our $\phi = 2$ results (solid line) and $\phi = 20$ results

Figure D.1: Welfare Superiority of $\phi = 2$ vs. $\phi = 20$



(dashed line), change for different degrees of industry concentration ranging from $n = 2$ to $n = 30$.

As the reader can see, the situation does not change much when ϕ increases by a factor of ten. While the region where OS is superior expands slightly for low values of γ and contracts slightly for γ near 1 the qualitative results change very little. (Notice that we have exaggerated the differences by drawing the figure so that the affected regions are magnified compared to the figure in our main text.) The reason for this similarity is that changes in ϕ affect the cost function for Pure-OS and Pure-CS industries identically. Furthermore, the values of γ where (a) $X^{os} = x^{cs}$ and (b) where x^{cs} has its minimum (and begins to rise again) are given by (a) $\gamma = {}^{2}/_{(n+2)}$ and (b) $\gamma = \left(n-5+\sqrt{9+n(n-2)}\right)/_{2(n-2)}$ and hence independent of ϕ.

D.4 Welfare of Pure vs. Mixed Cases in Concentrated Industries

Figures D.2 and D.3 show how our welfare analysis of Pure vs. Mixed cases changes for concentrated (small n) industries.

Figure D.2: Welfare-Comparison of Pure vs. Mixed Cases,$n = 30$

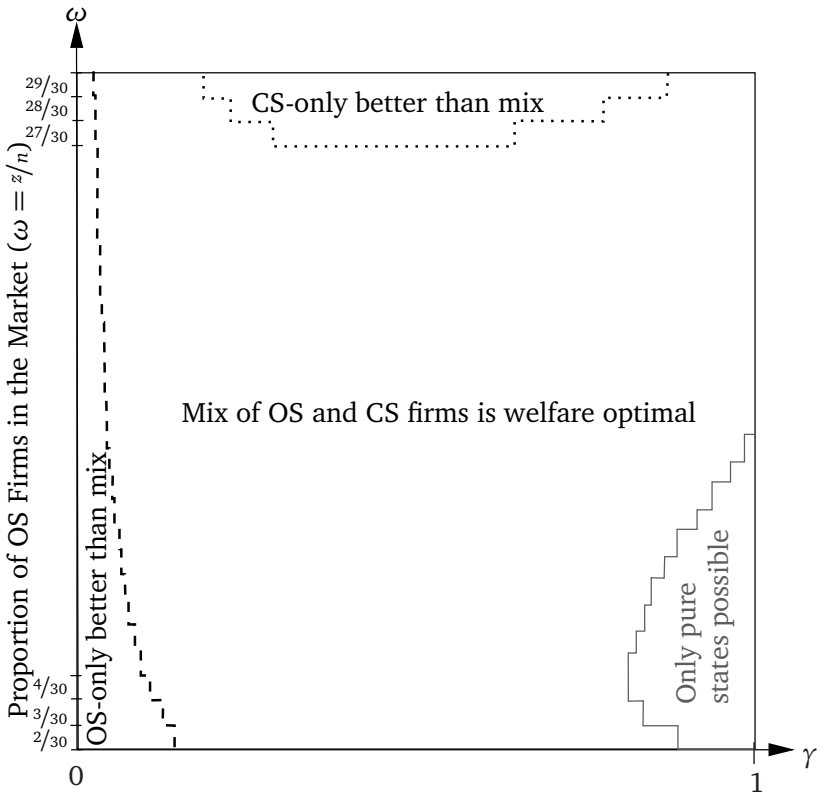
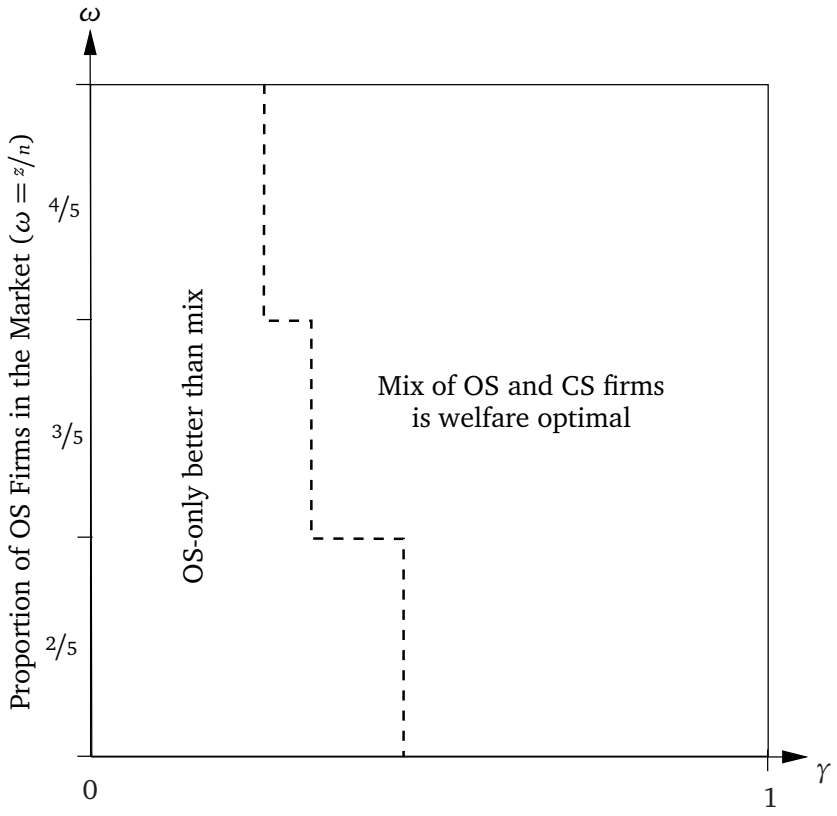


Figure D.3: Welfare-Comparison of Pure vs. Mixed Cases, $n = 5$



D.5 Alternative Motivation of OS-Entry into a Pure-CS Industry

Assume that an industry contains $n > 2$ incumbents, all of whom follow CS business models. Assume further that each incumbent's profits are zero so that further CS entry would produce negative profits. Finally, assume that there are $e \geq 2$ potential OS entrants. Each OS firm faces the following YES/NO choice on whether to enter.

For simplicity, we consider the case where $e = 2$ although our argument also holds for $e > 2$ entrants. Suppose that the $e = 2$ OS firms will earn a positive profit if they enter. Then the strategic problem is to ensure that both firms enter. (If only one OS firms enters, code-sharing cannot occur and the entrant will become a de facto CS firm earning negative profits.) This problem can be analyzed in terms of the following game where the payoffs have been normalized to 1, 0, and -1:

	YES	NO
YES	1,1	-1,0
NO	0,-1	0,0

This coordination game has two Nash-equilibria (YES, YES) and (NO, NO). Furthermore, this is a common interest game in which both two potential entrants would like to occupy the same (YES, YES) equilibrium. If players can signal which strategy they wish to play, we can assume that they will both arrive the common interest equilibrium. In our OS problem, this can readily be done if each OS entrant announces that its code will be subject to GPL.

E Appendix to Chapter 7

E.1 Descriptive Statistics of the Start-Up Analysis

Table E.1: Descriptive Statistics of the 2005 to 2008 Cohorts

Variable	Obs	Mean	Std. Dev.	Min	Max
oss	198	1.550505	1.43381	0	4
size (staff)	198	1.616919	1.141768	0.25	11
size (capital*)	135	1.217891	1.716922	0	8
lack of capital	198	0.3686869	0.4836718	0	1
convince financiers	198	0.2070707	0.4062338	0	1
Education	198	0.5808081	0.4946776	0	1
Experience	196	0.9030612	0.2966321	0	1
Necessity	198	0.510101	0.5011651	0	1
age of firm	198	4.050505	1.001255	2	5
new media & internet	198	0.4292929	0.49623	0	1
further dev. software	198	0.0808081	0.273231	0	1

* capital is in 10,000 EUR

Table E.2: Descriptive Statistics of the 2005 to 2007 Cohorts

Variable	Obs	Mean	Std. Dev.	Min	Max
oss	182	1.521978	1.451637	0	4
size (staff)	182	1.643681	1.175875	0.25	11
size (capital*)	123	1.274515	1.784314	0	8
lack of capital	182	0.3846154	0.4878463	0	1
convince financiers	182	0.2032967	0.4035616	0	1
Education	182	0.5659341	0.4970009	0	1
experience	180	0.9111111	0.2853771	0	1
necessity	182	0.5164835	0.5011068	0	1
age of firm	182	4.230769	0.8287236	3	5
new media & internet	182	0.4230769	0.4954103	0	1
further dev. software	182	0.0769231	0.2672044	0	1

* capital is in 10,000 EUR

Table E.3: Descriptive Statistics of the 2003 to 2006 Cohorts

Variable	Obs	Mean	Std. Dev.	Min	Max
oss	343	1.364431	1.430092	0	4
size (staff)	343	1.506851	1.072301	0.2	11
size (capital*)	260	1.185492	1.615155	0	8
lack of capital	343	0.3381924	0.4737855	0	1
convince financiers	343	0.1836735	0.3877834	0	1
Education	343	0.6355685	0.4819735	0	1
Experience	341	0.9208211	0.2704146	0	1
Necessity	343	0.3965015	0.4898854	0	1
age of firm	343	5.74344	1.016563	4	7
new media & internet	343	0.3877551	0.48795	0	1
further dev. software	343	0.0874636	0.2829259	0	1

* capital is in 10,000 EUR

E.2 Details on Variables of the Start-Up Analysis

Table E.4: Description of Variables

Variable	Coding	Explanation
oss	ordinal	Intensity of OSS-usage in the respective business field, at start-up time. The answering categories and the corresponding values are: 4 = "Yes, only", 3 = "Mainly", 2 = "About 50%", 1 = "To a small extent", 0 = "No (nearly) never".
size (staff)	real	Number of full positions – including the founders. We asked participants to count part-time positions regarding their percentage of full-time.
size (capital)	real	Sum of real capital and financial capital.
lack of capital	binary	1 if founders marked "Lack of own capital" as one of the start-up problems they faced (possibility to mark one or more problem).
convince financiers	binary	1 if founders marked "Difficulties in convincing potential financiers regarding the business concept" as one of the start-up problems they faced (possibility to mark one or more problem).
education	binary	1 if there was at least one person in the firm with an university diploma or other corresponding level of education (at start-up time).
experience	binary	1 if at least one of the founders had already experience in the sector.
necessity	binary	1 if start-up was necessity-based. Question: "Did you found the firm in order to realize a business idea or because there was no better alternative to generate income?" Answer categories are "To realize business idea", "There was no alternative way to generate income" (=necessity), "Because of both reasons", and "Other reasons, namely . . . "
age of firm	real	2010 minus the year of start-up. Question: "When did you start your business (Year of first turnover)"
new media & internet	binary	1 if "webhosting", or "web design/service" or "new media agency" is 1 (see list below)
further dev. software	binary	1 if "selfHW_furthSW " or "extHW_furthSW" or "furthSW " is 1 (see list below)

Table E.5: Description of Disaggregated Controls for the Business Fields

Variable	Coding	Explanation
Webhosting	binary	1 if "web hosting" is a start-up business field.
web design/service	binary	1 if "web design and web service" is a start-up business field.
new media agency	binary	1 if "services of (new media) agencies and related" is a start-up business field.
selfHW_extSW	binary	1 if "selling own hardware with third-party software" is a start-up business field.
selfHW_furthSW	binary	1 if "selling own hardware with further-developed software" is a start-up business field.
selfHW_selfSW	binary	1 if "selling own hardware with self-developed software" is a start-up business field.
extHW_extSW	binary	1 if "selling third-party hardware with third-party software" is a start-up business field.
extHW_furthSW	binary	1 if "selling third-party hardware with further-developed software" is a start-up business field.
extHW_selfSW	binary	1 if "selling third-party hardware with self-developed software" is a start-up business field.
extSW	binary	1 if "selling third-party software" is a start-up business field.
furthSW	binary	1 if "selling further-developed software" is marked sa start-up business field.
selfSW	binary	1 if "selling self-developed software" is a start-up business field.
Service_othSW	binary	1 if "service for software bought from a third-party" was a start-up business field.

Note: The category "third-party software" is defined as software developed by a third part, and the firm does not change the code before giving it as part of it's end-product to the customers. The category "third-party hardware" was distinguished from "own hardware" by the question whether the hardware was completely produced by a third party, or contains self-produced parts.

Bibliography

Acs, Z.J., Desai, S., Hessels, J., 2008. Entrepreneurship, economic development and institutions. Small Business Economics 31, 219 – 234.

Akerlof, G.A., Kranton, R.E., 2000. Economics and identity. The Quarterly Journal of Economics 115, 715–753.

Allarakhia, M., Kilgour, D.M., Fuller, J.D., 2010. Modelling the incentive to participate in open source biopharmaceutical innovation. R&D Management 40, 50 – 66.

Alverson, H., 1986. Culture and economy: Games that "play people". Journal of Economic Issues 20, 661–679.

Amoros, J.E., 2009. Entrepreneurship and Quality of Institutions. UNU-WIDER Research Papers 2009/07. World Institute for Development Economic Research (UNU-WIDER).

Asundi, J., 2005. The need for effort estimation models for open source software projects, in: Proceedings of the Fifth Workshop on Open Source Software Engineering, ACM, New York, NY, USA. pp. 1–3.

Au, Y.A., Carpenter, D., Chen, X., Clark, J.G., 2009. Virtual organizational learning in open source software development projects. Information & Management 46, 9 – 15.

Baake, P., Wichmann, T., 2004. Open Source Software, Competition and Potential Entry. Berlecon Research Papers 5. Berlecon Research. Berlin.

Bénabou, R., Tirole, J., 2006. Belief in a just world and redistributive politics. The Quarterly Journal of Economics 121, 699–746.

Benkler, Y., 2002. Coase's penguin, or, Linux and the nature of the firm. Yale Law Journal 112, 369–437.

Besen, S.M., Raskind, L.J., 1991. An introduction to the law and economics of intellectual property. Journal Of Economic Perspectives 5, 3–27.

Bessen, J., 2006. Open source software: Free provision of complex public goods, in: Bitzer, J., Schröder, P. (Eds.), The Economics Of Open Source Software Development. Elsevier, p. 57–81.

Bessen, J., Hunt, R.M., 2007. An empirical look at software patents. Journal of Economics & Management Strategy 16, 157–189.

Bessen, J.E., Hunt, R.M., 2004. The software patent experiment, in: OECD (Ed.), Patents, Innovation and Economic Performance. OECD, Paris.

den Besten, M., Dalle, J.M., Galia, F., 2008. The allocation of collaborative efforts in open-source software. Information Economics and Policy 20, 316 – 322.

Bisin, A., Verdier, T., 2001. The economics of cultural transmission and the dynamics of preferences. Journal of Economic Theory 97, 298–319.

Bitzer, J., 2004. Commercial versus open source software: The role of product heterogeneity in competition. Economic Systems 28, 369 – 381.

Bitzer, J., Schröder, P.J.H., 2007. Open source software, competition and innovation. Industry and Innovation 14, 461 – 476.

Blind, K., Edler, J., Friedewald, M., 2005. Software Patents – Economic Impacts and Policy Implications. Elgar, Cheltenham [u.a.].

Böhnlein, I., 2003. Anwendung von Aspekten der Neuen Institutionenökonomik auf Open Source Software. Produktion, Verfügungsrechte und Transaktionskosten – eine theoretische und empirische Untersuchung. Master's thesis. Johann Wolfgang Goethe-Universität. Frankfurt am Main.

Bonaccorsi, A., Giannangeli, S., Rossi, C., 2006. Entry strategies under competing standards: Hybrid business models in the open source software industry. Management Science 52, 1085–1098.

de Bondt, R., 1997. Spillovers and innovative activities. International Journal of Industrial Organization 15, 1–28.

Bosma, N., Fritsch, M., Schroeter, A., 2010. Institutions, capital and individual entrepreneurship recognition: an evolutionary conceptual model. Mimeo.

Bowles, S., 1998. Endogenous preferences: The cultural consequences of markets and other economic institutions. Journal of Economic Literature 36, 75–111.

Brand, A., Schmid, A., 2005. Koordination in einem Open Source-Projekt. Arbeitspapier aus dem DFG-Projekt Struktur und Funktionsweise elektronischer Arbeitsmärkte.

Brand, A., Schmid, A., 2006. Ist ein Open-Source-Projekt nur kooperativ? – Die Koordination der Zusammenarbeit im KDE-Projekt, in: Lutterbeck, B., Bärwolff, M., Gehring, R.A. (Eds.), Open Source Jahrbuch 2006: Zwischen Softwareentwicklung und Gesellschaftsmodell. Lehmanns Media, Berlin, pp. 123 – 138.

Brooks, F.P., 1982. The Mythical Man-Month: Essays On Software Engineering. Addison-Wesley.

Bulow, J.I., Geanakoplos, J.D., Klemperer, P.D., 1985. Multimarket oligopoly: Strategic substitutes and complements. The Journal of Political Economy 93, 488–511.

Burke, A., Fraser, S., 2007. The Impact of Intellectual Property Rights on Self-Employed Entrepreneurship: an International Analysis. Cranfield Management Research Paper Series 10/07. Cranfield University School of Management.

Business Software Alliance, 2007. Fifth annual BSA and IDC global software piracy study.

Campbell-Kelly, M., Garcia-Swartz, D.D., 2009. Pragmatism, not ideology: Historical perspectives on IBM's adoption of open-source software. Information Economics and Policy 21, 229 – 244.

Casadesus-Masanell, R., Ghemawat, P., 2006. Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows. Management Science 52, 1072–1084.

Casadesus-Masanell, R., Llanes, G., 2009. Mixed Source. Working Papers 09-06. NET Institute.

Ciffolilli, A., 2003. Phantom authority, self-selective recruitment and retention of members in virtual communities: The case of Wikipedia. First Monday 8. `http://firstmonday.org/`.

Coase, R.H., 1937. The nature of the firm. Economica 4, 386–405.

Comino, S., Manenti, F.M., Parisi, M.L., 2007. From planning to mature: On the success of open source projects. Research Policy 36, 1575 – 1586.

Cowan, R., Harison, E., 2001. Protecting the Digital Endeavour. Prospects for Intellectual Property Rights in the Information Society. Research Memoranda 28. MERIT, Maastricht Economic Research Institute on Innovation and Technology. Maastricht.

de Cremer, D., 1999. Trust and fear of exploitation in a public goods dilemma. Current Psychology , 153 – 163.

Crowston, K., Howison, J., 2005. The social structure of free and open source software development. First Monday 10. `http://firstmonday.org/`.

Crowston, K., Wei, K., Li, Q., Howison, J., 2006. Core and periphery in free/libre and open source software team communications, in: System Sciences: HICCS (Hawaii International Conference) Proceedings, pp. 118a–118a.

CSIS, 2008. Government open source policies. Center for Strategic and International Studies. Available at `http://csis.org/`.

Daffara, C., 2007. Business models in FLOSS-based software companies. Workshop presentation at the 3rd Conference on Open Source Systems. `http://opensource.mit.edu/papers/OSSEMP07-daffara.pdf`.

Dahlander, L., 2007. Penguin in a new suit: A tale of how de novo entrants emerged to harness free and open source software communities. Industrial and Corporate Change 16, 913 – 943.

Dahlander, L., Magnusson, M.G., 2005. Relationships between open source software companies and communities: Observations from nordic firms. Research Policy 34, 481 – 493.

Dahlander, L., Magnusson, M.G., 2006. Business models and community relationships of open source software firms, in: Bitzer, J., Schröder, P.J. (Eds.), The Economics of Open Source Software Development. Elsevier, Amsterdam, pp. 111 – 130.

Dahlander, L., Wallin, M.W., 2006. A man on the inside: Unlocking communities as complementary assets. Research Policy 35, 1243 – 1259.

Dalle, J.M., David, P.A., 2005. Allocation of software development resources in open source production mode, in: Feller, J., Fitzgerald, B., Hissam, S.A., Lakhani, K. (Eds.), Perspectives on Free and Open Source Software. Cambridge and London, pp. 297 – 328.

D'Antoni, M., Rossi, M.A., 2007. Copyright vs. Copyleft Licencing and Software Development. Department of Economics University of Siena Working Paper 510. Department of Economics, University of Siena.

d'Aspremont, C., Jacquemin, A., 1988. Cooperative and noncooperative r&d in duopoly with spillovers. American Economic Review 78, 1133–37.

David, P.A., Rullani, F., 2008. Dynamics of innovation in an 'open source' collaboration environment: Lurking, laboring, and launching floss projects on sourceforge. Industrial and Corporate Change 17, 647 – 710.

David, P.A., Waterman, A., Arora, S., 2003. FLOSS-US. The Free/Libre/Open Source Software Survey for 2003. Technical Report. Stanford Institute for Economic Policy Research.

Davis, L.E., North, D.C., 1971. Institutional Change and American Economic Growth. Cambridge University Press.

Demil, B., Lecocq, X., 2006. Neither market nor hierarchy nor network: The emergence of bazaar governance. Organization Studies 27, 1447 – 1466.

Demsetz, H., 1967. Towards a theory of property rights. American Economic Review , 347–359.

Deroian, F., Gannon, F., 2006. Quality-improving alliances in differentiated oligopoly. International Journal of Industrial Organization 24, 629–637.

Deshpande, A., Riehle, D., 2008. The total growth of open source, in: Russo, B., Damiani, E., Hissam, S., Lundell, B., Succi, G. (Eds.), Open Source Development, Communities and Quality. Springer. volume 275 of *IFIP International Federation for Information Processing*, pp. 197 – 209.

Diener, E., Gohm, C.L., Suh, E., Oishi, S., 2000. Similarity of the relation between marital status and subjective well-being across cultures. Journal of Cross-Cultural Psychology , 419–436.

Dixit, A., 1979. A model of duopoly suggesting a theory of entry barriers. Bell Journal of Economics 10, 20–32.

Economides, N., Katsamakas, E., 2006. Two-sided competition of proprietary vs. open source technology platforms and the implications for the software industry. Management Science 52, 1057–1071.

Edelman, B., 2002. Registrations in Open ccTLDs. Online Article. `cyber.law.harvard.edu/archived\_content/people/edelman/open-cctlds/`.

Eggertsson, T., 1990. Economic Behavior and Institutions. Cambridge University Press.

Eilhard, J., 2008. Loose contracts, tight control? Firms on SourceForge. CERNA Working Paper.

von Engelhardt, S., 2008a. The Economic Properties of Software. Jena Economic Research Papers 2008-045. Friedrich-Schiller-University Jena and Max-Planck-Institute of Economics.

von Engelhardt, S., 2008b. Intellectual Property Rights and Ex-Post Transaction Costs: the Case of Open and Closed Source Software. Jena Economic Research Papers 2008-047. Friedrich-Schiller-University Jena, Max-Planck-Institute of Economics.

von Engelhardt, S., 2010. Quality Competition or Quality Cooperation? License-Type and the Strategic Nature of Open Source vs. Closed Source Business Models. Jena Economic Research Papers 2010-034. Friedrich-Schiller-University Jena and Max-Planck-Institute of Economics.

von Engelhardt, S., Freytag, A., 2010. Institutions, Culture, and Open Source. Jena Economic Research Papers 2010-010. Friedrich-Schiller-University Jena and Max-Planck-Institute of Economics.

von Engelhardt, S., Freytag, A., Schulz, C., 2010. On the Geographic Allocation of Open Source Software Activities. Jena Economic Research Papers 2010-009. Friedrich-Schiller-University Jena and Max-Planck-Institute of Economics.

von Engelhardt, S., Maurer, S.M., 2010. The New (Commercial) Open Source: Does it Really Improve Social Welfare? Berkeley Goldman School of Public Policy Working Paper 10-001.

von Engelhardt, S., Pasche, M., 2004. Volkswirtschaftliche Aspekte Der Open-Source-Softwareentwicklung. Jenaer Schriften zur Wirtschaftswissenschaft.

von Engelhardt, S., Swaminathan, S., 2008. Open Source Software, Closed Source Software or Both: Impacts on Industry Growth and the Role of Intellectual Property Rights. Discussion Papers of DIW Berlin 799. DIW Berlin, German Institute for Economic Research.

Fernández, R., 2008. Culture and economics, in: Durlauf, S.N., Blume, L.E. (Eds.), The New Palgrave Dictionary of Economics. Palgrave Macmillan, 2nd edition.

Fernández, R., Fogli, A., 2009. Culture: An empirical investigation of beliefs, work, and fertility. American Economic Journal: Macroeconomics 1, 146–77.

Fershtman, C., Gandal, N., 2008. Microstructure of Collaboration: The 'Social Network' of Open Source Software. CEPR Discussion Papers 6789. C.E.P.R. Discussion Papers.

Fink, M., 2002. Business and Economics of Linux and Open Source. Prentice Hall Professional Technical Reference.

Fosfuri, A., Giarratana, M.S., Luzzi, A., 2008. The penguin has entered the building: The commercialization of open source software products. Organization Science 19, 292–305.

Foss, K., Foss, N., 2006. Entrepreneurship, transaction costs, and resource attributes. International Journal of Strategic Change Management 1, 53–60.

Franck, E., Jungwirth, C., 2003. Reconciling rent-seekers and donators–the governance structure of open source. Journal of Management and Governance 7, 401 – 421.

Freytag, A., Thurik, R. (Eds.), 2010. Entrepreneurship and Culture. Springer.

Fritsch, M., von Engelhardt, S., 2010. Who starts with open source? Institutional choice of German ICT start-ups. Mimeo.

Fritsch, M., Schroeter, A., 2009. Are More Start-Ups Really Better? Quantity and Quality of New Businesses and Their Effect on Regional Development. Jena Economic Research Papers 2009-070. Friedrich-Schiller-University Jena and Max-Planck-Institute of Economics.

Furubotn, E.G., Richter, R., 2005. Institutions and Economic Theory : The Contribution of the New Institutional Economics. Univ. Of Michigan Press.

Gaio, L., Rossi, A., den Besten, M., Dalle, J.M., 2009. Coordination, Division of Labor, and Open Content Communities: Template Messages in Wiki-Based Collections. DISA Working Papers 0903. Department of Computer and Management Sciences, University of Trento, Italy.

Gandal, N., 1994. Hedonic price indexes for spreadsheets and an empirical test of the network externalities hypothesis. RAND Journal Of Economics , 160–170.

Garzarelli, G., 2003. Open Source Software and The Economics Of Organization. Industrial Organization 0304003. EconWPA.

Gehring, R.A., 2006. The institutionalization of open source. Poiesis & Praxis: International Journal Of Technology Assessment and Ethics Of Science 4, 54–73.

Ghosh, R.A., 2006. Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU. Study. European Commission.

Ghosh, R.A., Glott, R., Krieger, B., Robles, G., 2002a. FLOSS Final Report, Part 3: Basics of Open Source Software Markets and Business Models. Technical Report. International Institute of Infonomics, University of Maastricht.

Ghosh, R.A., Glott, R., Krieger, B., Robles, G., 2002b. FLOSS Final Report, Part 4: Survey of Developers. Technical Report. International Institute of Infonomics, University of Maastricht.

Giuri, P., Ploner, M., Rullani, F., Torrisi, S., 2010. Skills, division of labor and performance in collective inventions: Evidence from open source software. International Journal of Industrial Organization 28, 54 – 68.

Giuri, P., Rocchetti, G., Torrisi, S., 2002. Open Source Software: From Open Science to New Marketing Models. LEM Papers Series 2002/23. Laboratory of Economics and Management (LEM), Sant'Anna School of Advanced Studies, Pisa, Italy.

Giuri, P., Rullani, F., Torrisi, S., 2008. Explaining leadership in virtual teams: The case of open source software. Information Economics and Policy 20, 305 – 315.

Gonzalez-Barahona, J.M., Robles, G., Andradas-Izquierdo, R., Ghosh, R.A., 2008. Geographic origin of libre software developers. Information Economics and Policy 20, 356 – 363.

Graham, S., Somaya, D., 2004. The use of patents, copyrights and trademarks in software: Evidence from litigation, in: OECD (Ed.), Patents, Innovation and Economic Performance. OECD, Paris.

Greene, W., 2008. Econometric Analysis. Pearson Prentice Hall. 6. ed. edition.

Greif, A., 1994. Cultural beliefs and the organization of society: A historical and theoretical reflection on collectivist and individualist societies. Journal of Political Economy 102, 912–50.

Gröhn, A., 1999. Netzwerkeffekte und Wettbewerbspolitik. Eine Ökonomische Analyse Des Softwaremarktes. Mohr Siebeck, Tübingen.

Gruber, M., Henkel, J., 2006. New ventures based on open innovation – an empirical analysis of start-up firms in embedded Linux. International Journal of Technology Management 33, 356–372.

Guiso, L., Sapienza, P., Zingales, L., 2006. Does culture affect economic outcomes? Journal of Economic Perspectives 20, 23–48.

Guiso, L., Sapienza, P., Zingales, L., 2008. Social capital as good culture. Journal of the European Economic Association 6, 295–320.

Gwartney, J., Lawson, R., Norton, S., 2008. Economic Freedom of the World – 2008 Annual Report. The Fraser Institute. Data retrieved from `http://www.freetheworld.com/`.

Häckner, J., 2000. A note on price and quantity competition in differentiated oligopolies. Journal of Economic Theory 93, 233–239.

Hall, B.H., MacGarvie, M., 2006. The Private Value of Software Patents. NBER Working Papers 12195. National Bureau of Economic Research, Inc.

Hall, J.C., Sobel, R.S., 2008. Institutions, entrepreneurship, and regional differences in economic growth. Southern Journal of Entrepreneurship 1, 69–96.

Hardin, G., 1968. The tragedy of the commons. Science 162, 1243 – 1248.

Harison, E., Cowan, R., 2004. On substitution of intellectual property and free disclosure: An analysis of R&D strategies in software technologies. Economics of Innovation and New Technology 13, 477 – 487.

Harison, E., Koski, H., 2010. Applying open innovation in business strategies: Evidence from finnish software firms. Research Policy 39, 351–359.

Hars, A., Ou, S., 2002. Working for free? - Motivations of participating in open source projects. International Journal of Electronic Commerce 6, 25 – 39.

Hart, O., Moore, J., 1990. Property rights and the nature of the firm. Journal of Political Economy 98, 1119–1158.

Hawkins, R.E., 2004. The economics of open source software for a competitive firm: Why give it away for free? Netnomics 6, 103 – 117.

von Hayek, F.A., 1967. Studies in Philosophy, Politics and Economics. University of Chicago Press. chapter The Results of Human Action but not of Human Design. pp. 96–105.

von Hayek, F.A., 1975. The pretence of knowledge. Swedish Journal of Economics 77, 433 – 442.

von Hayek, F.A., 1978. New Studies in Philosophy, Politics, Economics, and the History of Ideas. University of Chicago Press. chapter Competition as a Discovery Procedure. pp. 179–190.

Heller, M., 1998. The tragedy of the anticommons: Property in the transition from marx to markets. Harvard Law Review , 621–688.

Henkel, J., 2004. The jukebox mode of innovation - a model of commercial open source development .

Henkel, J., 2006. Selective revealing in open innovation processes: The case of embedded linux. Research Policy 35, 953 – 969.

Henkel, J., 2009. Champions of revealing–the role of open source developers in commercial firms. Industrial and Corporate Change 18, 435 – 471.

Henkel, J., von Hippel, E., 2005. Welfare implications of user innovation. The Journal of Technology Transfer 30, 73–87.

Henkel, J., Maurer, S.M., 2007. The economics of synthetic biology. Molecular Systems Biology 3.

Henrekson, M., 2007. Entrepreneurship and institutions. Comparative Labor Law & Policy Journal 28, 717–742.

Henrekson, M., Sanandaji, T., 2010. The Interaction of Entrepreneurship and Institutions. Working Paper Series 830. Research Institute of Industrial Economics.

Henrich, J., 2000. Does culture matter in economic behavior? Ultimatum game bargaining among the machiguenga of the peruvian amazon. American Economic Review 90, 973–979.

Hertel, G., Niedner, S., Herrmann, S., 2003. Motivation of software developers in open source projects: an internet-based survey of contributors to the linux kernel. Research Policy 32, 1159 – 1177.

Heydemann, S., 2008. Institutions and economic performance: The use and abuse of culture in new institutional economics. Studies in Comparative International Development 43, 27–52.

von Hippel, E., 2005. Open source software projects as user innovation networks, in: von Hippel, E., Feller, J., Fitzgerald, B., Hissam, S.A., Lakhani, K.R. (Eds.), Perspectives on Free and Open Source Software. MIT Press, Cambridge and London, pp. 267 – 278.

von Hippel, E., von Krogh, G., 2003. Open source software and the 'private-collective' innovation model: Issues for organization science. Organization Science 14, 209 – 223.

Hofstede, G.H., 1991. Cultures and organizations: Software of the mind. McGraw-Hill, London and New York.

Holmes, T.J., Schmitz, James A, J., 1990. A theory of entrepreneurship and its application to the study of business transfers. Journal of Political Economy 98, 265–94.

Hope, J., 2008. Biobazaar: The Open Source Revolution and Biotechnology. Harvard University Press.

Hsu, J., Wang, X., 2005. On welfare under cournot and bertrand competition in differentiated oligopolies. Review of Industrial Organization 27, 185–191.

Iannacci, F., Mitleton-Kelly, E., 2005. Beyond markets and firms: The emergence of Open Source networks. First Monday 10. `http://firstmonday.org/`.

International Telecommunication Union, 2006. ICT Statistics 2006, Part 4 – Internet Indicators: subscribers, users and broadband subscribers. ICT Statistics. ITU. `http://www.itu.int/ITU-D/ICTEYE/Reports.aspx`.

Jackson, W.A., 1993. Culture, society and economic theory. Review of Political Economy 5, 453–469.

Jensen, C., Scacchi, W., 2007. Role migration and advancement processes in OSSD projects, in: International Conference on Software Engineering, Minneapolis, MN, USA.

Johnson, J.P., 2002. Open source software: Private provision of a public good. Journal of Economics & Management Strategy 11, 637–662.

Johnson, J.P., 2006. Collaboration, peer review and open source software. Information Economics and Policy 18, 477 – 497.

Wendel de Joode, R.V., Bruijn, J.A.d.d., Eeten, M.J.G.V., 2003. Protecting the Virtual Commons – Self-Organizing Open Source and Free Software Communities and Innovative Intellectual Property Regimes. T.M.C. Asser Press, The Hague.

Kahin, B., 2004. Through the lens of intangibles: What patents on software and services reveal about the system, in: OECD (Ed.), Patents, Innovation and Economic Performance. OECD, Paris.

Katz, M.L., Shapiro, C., 1994. Systems competition and network effects. (symposia network externalities). Journal of Economic Perspectives, 8, 93–115.

Kihlstrom, R.E., Laffont, J.J., 1979. A general equilibrium entrepreneurial theory of firm formation based on risk aversion. Journal of Political Economy 87, 719–48.

Knight, F.H., 1921. Risk, Uncertainty and Profit. Hart, Schaffner, and Marx.

Koch, S., 2004. Profiling an open source project ecology and its programmers. Electronic Markets 14, 77–88.

Kooths, S., Langenfurth, M., Kalwey, N., 2003. Open-Source Software: An Economic Assessment. MICE Economic Research Studies 4. Muenster Institute For Computational Economics. Münster.

Koski, H., 2005. OSS production and licensing strategies of software firms. Review of Economic Research on Copyright Issues 2, 111 – 125.

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

Kugler, P., 2005. Coordinating Innovation: Evidence from Open Source Software Development. Difo-Druck GmbH.

Kumar, S., 2006. Enforcing the GNU GPL. Journal of Law, Technology & Policy 1.

de Laat, P.B., 2005. Copyright or copyleft?: An analysis of property regimes for software development. Research Policy 34, 1511 – 1532.

de Laat, P.B., 2007. Governance of open source software: State of the art. Journal of Management and Governance 11, 165 – 177.

Lakhani, K.R., Wolf, B., Bates, J., DiBona, C., 2002. The boston consulting group/OSDN hacker survey. `http://www.osdn.com/bcg/`.

Lakhani, K.R., Wolf, R.G., 2005. Why hackers do what they do: Understanding motivation and effort in free/open source software projects, in: Feller, J., Fitzgerald, B., Hissam, S.A., Lakhani, K.R. (Eds.), Perspectives on Free and Open Source Software. MIT Press, pp. 3 –22.

Lamastra Rossi, C., 2009. Software innovativeness. A comparison between proprietary and free/open source solutions offered by italian SMEs. R&D Management 39, 153 – 169.

Lancashire, D., 2001. Code, culture and cash: The fading altruism of open source development. First Monday 6. `http://firstmonday.org`.

Langlois, R.N., 2002. Modularity in technology and organization. Journal Of Economic Behavior & Organization 49, 19–37.

Langlois, R.N., Garzarelli, G., 2008. Of hackers and hairdressers: Modularity and the organizational economics of open-source collaboration. Industry and Innovation 15, 125 – 143.

Lattemann, C., Stieglitz, S., 2005. Framework for governance in open source communities, in: Proceedings of the 38th Annual Hawaii International Conference on System Sciences 2005, pp. 192–202.

Lee, G.K., Cole, R.E., 2003. From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development. Organization Science 14, 633–649.

Lerner, J., Pathak, P.A., Tirole, J., 2006. The dynamics of open-source contributors. American Economic Review 96, 114.

Lerner, J., Tirole, J., 2000. The Simple Economics of Open Source. NBER Working Paper Series 7600.

Lerner, J., Tirole, J., 2002. Some simple economics of open source. Journal of Industrial Economics 50, 197–234.

Lerner, J., Tirole, J., 2005. The scope of open source licensing. Journal of Law, Economics and Organization 21, 20–56.

Lerner, J., Zhu, F., 2007. What is the impact of software patent shifts? Evidence from lotus v. borland. International Journal of Industrial Organization 25, 511–529.

Lessig, L., 1999. Code and Other Laws Of Cyberspace. Basic Books, New York, NY.

Lessig, L., 2006. Code: Version 2.0. Basic Books.

Libecap, G.D. (Ed.), 2004. Intellectual Property and Entrepreneurship. volume 15 of *Advances in the Study of Entrepreneurship, Innovation & Economic Growth*. JAI Press.

Liebowitz, S.J., Margolis, S.E., 1994. Network externality: An uncommon tragedy. Journal Of Economic Perspectives 8, 133–150.

Liebowitz, S.J., Margolis, S.E., 2002. Network effects, in: Cave, M., Majumdar, S., Vogelsang, I. (Eds.), Handbook of Telecommunications Economics. volume 1, pp. 76–94.

Llanes, G., de Elejalde, R., 2009. Industry Equilibrium with Open Source and Proprietary Firms. Harvard Business School Working Papers 09-149. Harvard Business School.

Lucas, R.E., 1978. On the size distribution of business firms. Bell Journal of Economics 9, 508–523.

Madey, G., n. d. The SourceForge research data archive (SRDA). `http://srda.cse.nd.edu`.

Markus, M.L., 2007. The governance of free/open source software projects: Monolithic, multidimensional, or configurational? Journal of Management and Governance 11, 151 – 163.

Maurer, S.M., 2008. Open source biology: Finding a niche (or maybe several). UMKC Law Review 76.

Maurer, S.M., Scotchmer, S., 2004. Procuring knowledge, in: Libecap, G.D., Thursby, M.C. (Eds.), Intellectual Property and Entrepreneurship. Emerald Group Publishing. Advances in the Study of Entrepreneurship, Innovation, and Economic Growth, pp. 1–31.

Maurer, S.M., Scotchmer, S., 2006. Open source software: the new intellectual property paradigm, in: Hendershott, T. (Ed.), Handbook of Economics and Information Systems. Elsevier, pp. 285–319.

McGowan, D., 2001. The legal implications of open source software. Illinois Law Review , 241–304.

Mustonen, M., 2001. Copyleft - The Economics Of Linux and Other Open Source Software. Discussion Papers of the Department of Economics, University of Helsinki 493. University of Helsinki.

Mustonen, M., 2003. Copyleft–the economics of linux and other open source software. Information Economics and Policy 15, 99 – 121.

Myerson, R.B., 1991. Game Theory: Analysis of Conflict. Harvard University Press.

Nguyen, T.V., Bryant, S.E., Rose, J., Tseng, C.H., Kapasuwan, S., 2009. Cultural values, market institutions, and entrepreneurship potential: A comparative study of the united states, taiwan, and vietnam. Journal of Developmental Entrepreneurship (JDE) 14, 21–37.

Norris, P., 2009. Democracy crossnational data, release 3.0 – spring 2009. Dataset available at `http://www.hks.harvard.edu/fs/pnorris/Data/Data.htm`.

North, D.C., 1990. Institutions, Institutional Change and Economic Performance. Cambridge Univ. Press, Cambridge.

North, D.C., 1994. Economic performance through time. American Economic Review 84, 359–368.

Nugent, J.B., 2008. Institutions and economic performance: The use and abuse of culture in new institutional economics: A response to Heydemann. Studies in Comparative International Development 43, 206–217.

Nüttgens, M., Tesei, E., 2000. Open Source: Marktmodelle und Netzwerke. Veröffentlichungen des Instituts für Wirtschaftsinformatik. Saarbrücken.

Nyström, K., 2008. The institutions of economic freedom and entrepreneurship: evidence from panel data. Public Choice 136, 269–282.

Oishi, S., 2000. Goals as cornerstones of subjective well-being: Linking individuals with cultures, in: Diener, E., Suh, E. (Eds.), Cross-Cultural Psychology of Subjective Well-Being. MIT Press, pp. 87 – 112.

Olson, Mancur, J., Zeckhauser, R., 1966. An economic theory of alliances. The Review of Economics and Statistics 48, 266–279.

Olson, M., 1971. The Logic Of Collective Action. Harvard Univ. Press, Cambridge, Mass.

O'Mahony, S., 2003. Guarding the commons: How community managed software projects protect their work. Research Policy 32, 1179 – 1198.

O'Mahony, S., Ferraro, F., 2007. The emergence of governance in an open source community. Academy of Management Journal 50, 1079 – 1106.

Osterloh, M., Rota, S., 2007. Open source software development–just another case of collective invention? Research Policy 36, 157 – 171.

Osterloh, M., Rota, S., von Wartburg, M., 2001. Open Source - New Rules in Software Development. Technical Report.

Ostrom, E., 1998. A behavioral approach to the rational choice theory of collective action. American Political Science Review , 1 – 22.

Pais, A., 1983. Subtle is the Lord: The Science and Life of Albert Einstein. Oxford University Press.

Parker, S., 2009. The Economics of Entrepreneurship. Cambridge University Press.

Pejovich, S., 2003. Understanding the transaction costs of transition: it's the culture, stupid. The Review of Austrian Economics 16, 347–361.

Perkins, G., 1998. Open source and capitalism. Online article posted at slashdot.org: `http://slashdot.org/articles/980824/0854256.shtml`.

Picot, A., Dietl, H., Franck, E., 2005. Organisation: eine ökonomische Perspektive. Schäffer-Poeschel. 4., überarb. und erw. aufl edition.

Pilch, H., 2004. Why are software patents so trivial?, in: OECD (Ed.), Patents, Innovation and Economic Performance. OECD, Paris.

Pisano, G., 2006. Profiting from innovation and the intellectual property revolution. Research Policy 35, 1122–1130.

Polanski, A., 2007. Is the General Public Licence a rational choice? Journal of Industrial Economics 55, 691–714.

Putnam, R.D., 1993a. Bowling alone: America's declining social capital. Journal of Democracy 6, 65–78.

Putnam, R.D., 1993b. The prosperous community: Social capital and public life. American Prospect 4, 35–42.

Quah, D., 2003. Digital Goods and The New Economy. CEP Discussion Papers 563. London School of Economics. London.

Rabin, M., 1993. Incorporating fairness into game theory and economics. American Economic Review 83, 1281–1302.

Ramanujam, P., 2007. Who Produces Open Source Software. MPRA Paper 4253. University Library of Munich, Germany.

Raymond, E.S., 1998. The cathedral and the bazaar. First Monday 3. `http://firstmonday.org`.

Riehle, D., 2007. The economic motivation of open source software: Stakeholder perspectives. IEEE Computer 40, 25 – 32.

Robles, G., Gonzalez-Barahona, J.M., 2006. Geographic location of developers at SourceForge, in: MSR '06: Proceedings of the 2006 international workshop on Mining software repositories, ACM, New York, NY, USA. pp. 144–150.

Robles, G., Scheider, H., Tretkowski, I., Weber, N., 2001. Who is doing it? A research on libre software developers. Technical Report. Technische Universität Berlin.

Romberg, T., 2003. Herstellerübergreifende Wiederverwendung von Komponenten, in: Handbuch Zur Komponentenbasierten Softwareentwicklung. Frauenhofer-Institut Für Experimentelles Software Engineering / Forschungszentrum Informatik.

Roosendaal, A., 2007. Biomedics and creative commons: A perfect match? International Journal of Intellectual Property Management 1, 316 – 324.

Rosenbaum, E.F., 2001. Culture, cognitive models, and the performance of institutions in transformation countries. Journal of Economic Issues 35, 889–909.

Rossi, C., Bonaccorsi, A., 2006. Intrinsic motivations and profit-oriented firms in open source software: Do firms practise what they preach?, in: Schröder, J.B., H., P.J. (Eds.), The Economics Of Open Source Software Development. Elsevier, pp. 84–109.

Rossi, M.A., 2006. Decoding the free/open source software puzzle: A survey of theoretical and empirical contributions, in: Bitzer, J., Schröder, P. (Eds.), The Economics Of Open Source Software Development. Elsevier, pp. 15–55.

Sadowski, B.M., Sadowski-Rasters, G., Duysters, G., 2008. Transition of governance in a mature open software source community: Evidence from the debian case. Information Economics and Policy 20, 323 – 332.

Schmidt, K.M., Schnitzer, M., 2003. Public subsidies for open source? Some economic policy issues of the software market. Harvard Journal of Law & Technology 16.

Schmidtke, R., 2006. Private Provision of a Complementary Public Good. CESifo Working Paper Series CESifo Working Paper No. 1756. CESifo Group Munich.

Schweik, C.M., Semenov, A., 2003. The institutional design of open source programming: Implications for adressing complex public policy and management problems. First Monday 8. `http://firstmonday.org/`.

Scotchmer, S., 2004. Innovation and Incentives. The MIT Press.

Sen, R., 2007. A strategic analysis of competition between open source and proprietary software. Journal of Management Information Systems 24, 233 – 257.

Sen, R., Subramaniam, C., Nelson, M.L., 2008. Determinants of the Choice of Open Source Software License. Journal of Management Information Systems 25, 207 – 239.

Shy, O., 2001. The Economics of Network Industries. Cambridge University Press, Cambridge.

Simcoe, T.S., Graham, S.J.H., Feldman, M.P., 2009. Competing on standards? Entrepreneurship, intellectual property, and platform technologies. Journal of Economics and Management Strategy 18, 775–816.

Singh, N., Vives, X., 1984. Price and quantity competition in a differentiated duopoly. RAND Journal of Economics 15, 546–554.

Stallman, R.M., 1999. The GNU operating system and the free software movement, in: DiBona, C., Ockman, S., Stone, M. (Eds.), Open Sources: Voices From The Open Source Revolution. O'Reilly Media, Inc., pp. 53–70.

Stam, W., 2009. When does community participation enhance the performance of open source software companies? Research Policy 38, 1288 – 1299.

Steinmueller, W.E., 1996. The U.S. software industry: An analysis and interpretive history, in: Mowery, D.C. (Ed.), The International Computer Software Industry: A Comparative Study of Industry Evolution and Structure. Oxford University Press, Oxford, New York, pp. 15–51.

Suh, E., Diener, E., Oishi, S., Triandis, H.C., 1998. The shifting basis of life satisfaction judgments across cultures: Emotions versus norms. Journal of Personality and Social Psychology 74, 482 – 493.

Sutton, J., 1997. One smart agent. RAND Journal of Economics 28, 605–628.

Symeonidis, G., 2003. Comparing cournot and bertrand equilibria in a differentiated duopoly with product R&D. International Journal of Industrial Organization 21, 39–55.

Tabellini, G., 2007. Culture and Institutions: Economic Development in the Regions of Europe. Levine's Working Paper Archive 321307000000000974. David K. Levine.

Tabellini, G., 2008a. Institutions and culture. Journal of the European Economic Association 6, 255–294.

Tabellini, G., 2008b. The scope of cooperation: Values and incentives. The Quarterly Journal of Economics 123, 905–950.

UNDP, 2004. Human Development Report 2004. United Nations Development Programme.

Välimäki, M., 2003. Dual licensing in open source software industry. Systemes d'Information et Management 8, 63 – 75.

Verani, S., 2006. Open Source Development in a Differentiated Duopoly. Economics Discussion / Working Papers 06-05. The University of Western Australia, Department of Economics.

Weber, K., 2004a. Social aspects of non-proprietary software. International Review of Information Ethics 2.

Weber, S., 2000. The Political Economy of Open Source Software. UCAIS Berkeley Roundtable on the International Economy, Working Paper Series 1011. UCAIS Berkeley Roundtable on the International Economy, UC Berkeley.

Weber, S., 2004b. The Success Of Open Source. Harvard University Press.

West, J., Gallagher, S., 2006. Challenges of open innovation: the paradox of firm investment in open-source software. R&D Management 36, 319 – 331.

West, J., O'Mahony, S., 2008. The role of participation architecture in growing sponsored open source communities. Industry and Innovation 15, 145 – 168.

von Westarp, F., 2003. Modeling Software Markets. Physica-Verlag, Heidelberg.

White, A.G., Abel, J.R., Berndt, E.R., Monroe, C.W., 2005. Hedonic price indexes for personal computer operating systems and productivity suites. Annales d'Economie et de Statistique 79/80, 247–260.

Williamson, C.R., 2009. Informal institutions rule: institutional arrangements and economic performance. Public Choice 139, 371–387.

Williamson, O.E., 2000. The new institutional economics: Taking stock, looking ahead. Journal of Economic Literature 38, 595–613.

World Bank, 2007. World Development Indicators 2007.

World Values Survey Association, n.d. World values survey – online dataset. Available at `www.worldvaluessurvey.org`.

Wynn, D.E., 2003. Organizational structure of open source projects: A life cycle approach, in: Proceedings of the 7th Annual Conference of the Southern Association for Information Systems, pp. 285–290.

Xu, J., Christley, S., Madey, G., 2006. Application of social network analysis to the study of open source software, in: Bitzer, J., Schröder, P.J. (Eds.), The Economics of Open Source Software Development. Elsevier Press.

Yamagishi, T., Kanazawa, S., Mashima, R., Terai, S., 2005. Separating trust from cooperation in a dynamic relationship: Prisoner's dilemma with variable dependence. Rationality and Society 17, 275 – 308.

Yamagishi, T., Yamagishi, M., 1994. Trust and commitment in the united states and japan. Motivation and Emotion 18, 129 – 166.

Erklärung

Hiermit erkläre ich,

1.) dass mir die geltende Promotionsordnung bekannt ist;

2.) dass ich die Dissertation selbst angefertigt, keine Textabschnitte eines Dritten oder eigener Prüfungsarbeiten ohne Kennzeichnung übernommen und alle von mir benutzten Hilfsmittel, persönlichen Mitteilungen und Quellen in meiner Arbeit angegeben habe;

3.) dass ich bei der Auswahl und Auswertung des Materials sowie bei der Herstellung des Manuskriptes keine unzulässige Hilfe in Anspruch genommen habe;

4.) dass ich nicht die Hilfe eines Promotionsberaters in Anspruch genommen habe und dass Dritte weder unmittelbar noch mittelbar geldwerte Leistungen von mir für Arbeiten erhalten haben, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen;

5.) dass ich die Dissertation noch nicht als Prüfungsarbeit für eine staatliche oder andere wissenschaftliche Prüfung eingereicht habe;

6.) dass ich nicht die gleiche, eine in wesentlichen Teilen ähnliche oder eine andere Abhandlung bei einer anderen Hochschule bzw. anderen Fakultät als Dissertation eingereicht habe.