Dynamics of Institutions and Markets in Europe is a network of excellence of social scientists in Europe, working on the economic and social consequences of increasing globalization and the rise of the knowledge economy.
http://www.dime-eu.org/

DIME Working Papers on
INTELLECTUAL PROPERTY RIGHTS

http://www.dime-eu.org/working-papers/wp14

Sponsored by the
6th Framework Programme
of the European Union

Emerging out of DIME Working Pack:

'The Rules, Norms and Standards on Knowledge Exchange'

Further information on the DIME IPR research and activities:

http://www.dime-eu.org/wp14

This working paper is submitted by:

Stefano Comino[*] and Fabio M. Manenti[**]

[*]Dipartimento di Economia, Universit`a di Trento
Email: stefano.comino@economia.unitn.it
[**]Dipartimento di Scienze Economiche "M. Fanno", Universit`a di Padova,
Email: fabio.manenti@unipd.it

Dual Licensing in Open Source Software Markets

**This is Working Paper
No 31 (April 2008)**

The Intellectual Property Rights (IPR) elements of the DIME Network currently focus on research in the area of patents, copyrights and related rights. DIME's IPR research is at the forefront as it addresses and debates current political and controversial IPR issues that affect businesses, nations and societies today. These issues challenge state of the art thinking and the existing analytical frameworks that dominate theoretical IPR literature in the fields of economics, management, politics, law and regulation- theory.

Dual Licensing in Open Source Software Markets*

Stefano Comino[†] Fabio M. Manenti[‡]

March 2008

Abstract

In this paper we present a theoretical model to study the characteristics and the commercial sustainability of a particular open source strategy known as dual licensing. We show that the decision to employ a dual licensing strategy occurs whenever the the strength and the relevance of the contribution of the OS community is sufficiently large. This result points to the crucial role of OS licensing schemes for firms embracing open source strategies, and can help explaining the observed proliferation of the open source licenses.

J.E.L. codes: L11, L17, L86, D45.

Keywords: open source software, dual licensing, copyright, versioning, license proliferation.

---

*Much of this work has been conducted while the authors were visiting the School of Information of the University of California at Berkeley. We are extremely grateful to Hal Varian for his hospitality. Financial support from "Progetto di Ateneo" - Padova, 2005.

[†]Dipartimento di Economia, Università di Trento, Via Inama 5, 38100 TRENTO (Italy), Tel. (39) 0461 882221, email: `stefano.comino@economia.unitn.it`

[‡]Corresponding author: Dipartimento di Scienze Economiche "M. Fanno", Università di Padova, Via del Santo 33, 35123 PADOVA (Italy), Tel. (39) 049 8274238, Fax. (39) 049 8274211, email: `fabio.manenti@unipd.it`

1 Introduction

Until recently, open source (OS) has been seen unfamiliar by the business community and, in many cases, it has been perceived as a real threat by commercial vendors. In the very last years, things have changed substantially and both large established incumbents such as IBM, HP or NEC as well as start-ups are increasingly embracing OS strategies (Hecker, 1999; Bonaccorsi et al., 2006; Rajala et al., 2007).

Commercial firms may enjoy several benefits by "going open source". For instance, a firm may take advantage of the contributions of the community of OS developers either in the direct form of code enhancements or in terms of educated feedbacks and reviews received from expert users. Furthermore, open source represents a powerful channel of software distribution: it may constitute a key strategic instrument to improve the perceived quality of the product and to enlarge the installed base of users, thus helping establishing an industry standard. Also, the decision to release the source code to the OS community might be necessary in order to cope with an increased competition in the marketplace.[1]

The key issue for a software vendor is how to design a sustainable business model based on open source solutions, provided that various features of OS software development and distribution seem to be unappropriate for commercial exploitation.[2] A typical commercial strategy that has been successfully employed is to sell complementary products and to profit from OS-related segments and/or services.[3]

In this paper we focus on an alternative business model that is gaining popularity among

[1]According to many commentators, Netscape started the Mozilla project due to its inability to meet the competitive threat posed by Internet Explorer; see West and Gallagher (2004).

[2]For instance, OS licenses require the code of the software to be freely re-distributable; when releasing the software code, an individual, or the firm cannot prevent or restrict (e.g. by requiring royalties) its re-distribution. More specifically, article 1 of the Open Source definition states that "The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale." (See the OSI definition WEB site: `www.opensource.org/docs/definition.php`).

[3]Just to take a relevant example, in 2001, IBM started the open source project Eclipse in order to promote the use of the programming language Java within server products; IBM profited from selling related products such as components of WebSphere and WebLogic (West and Gallagher, 2004). Alternatively, software vendors often offer deployment support, customization and adds-on products for OS solutions; see Rajala et al. (2007) for a comprehensive discussion of the well-known Red Hat case.

software vendors which is known as *dual licensing*. With dual licensing firms mix traditional and OS-based strategies by offering their software product under both a traditional proprietary license and an open source one;[4] in the latter case, the software is typically provided for free or at a nominal fee. There are various reasons why customers, when offered a free OS version of a software, may still prefer to pay for the proprietary version; certainly, the most important reason accrues from the reciprocal provision usually imposed by OS licenses: open source customers are required to redistribute their derived works under the same licensing scheme as the original software, including the requirement to make the source code of the derived software publicly available. To better grasp this critical issue, it is useful to quote Michael Olsen, CEO of Sleepycat Software Inc., producer of the embedded database BerkeleyDB; Olsen describes the dual licensing strategy of his company as follows:

> *"The Sleepycat open source license permits you to use Berkeley DB [...] at no charge under the condition that if you use the software in an application you redistribute, the complete source code for your application must be available and freely redistributable under reasonable conditions. If you do not want to release the source code for your application, you may purchase a license from Sleepycat Software."* (Välimäki, 2005 pp. 209-210)

Customers that use, modify and embed Berkeley DB into their own aplications might be reluctant to use the OS version. These applications may be products per se or, more frequently, they may constitute part of a more complex system that customers produce and sell. In both instances, it is clear that since customers want to keep proprietary control on their derived product, they may be unhappy with the reciprocal provision of the Sleepycat open source license; in this case, customers may prefer to pay for the proprietary version in order to be relieved from this provision.[5] Consequently, the adoption of a dual licensing

[4]Välimäki (2005) presents in details three well known examples of companies that have succesfully adopted a dual licensing strategy: Sleepycat, MySQL and Trolltech. On the Sleepycat experience see also Olson (2006). According to a recent paper by Koski (2007), nearly 6% out of a sample of 270 OS software business firms in Finland, Germany, Italy, Portugal and Spain, employ dual licensing strategies.

[5]This explains the fears of the many companies that, for example, embed OS Linux into their own platforms but that decide not to release their derived code under the GPL, because it represents the bulk of their product's core technology. They fear that some day a court may oblige them to adhere to the GPL reciprocal provision and to release their derived software to the public domain. This fear has been exacerbated since the well known "SCO vs IBM" case, whereby IBM has been alleged to violating SCO's intellectual property rights by distributing a Linux distribution with copied code; see Moglen (2003).

strategy may allow the software vendor to benefit from going open source, without, at the same time, sacrificing commercial opportunities.

This is the basic story of dual licensing which hinges upon the role of reciprocity of OS licenses; noteworthy, there are other reasons that might make the proprietary version of the software preferred to the free OS version. Customers may be willing to receive products with more warranties or less limitations of the liability of the licensor than those usually available for OS licensed products; or they might want to receive technical or aftersale support from the software vendor which is not normally offered to those that adopt the open source version. Also, in many instances, OS licenses allow the licensor to terminate the agreement conditional on the occurrence of specific events,[6] and this clearly puts the customer to a risk in case she/he needs to invest money and effort in using the software. Finally, the proprietary version may come with additional features and functionalities that are of particular value to customers.

Our paper offers also a possible explantation to an interesting phenomenon that has been characterizing the OS world in the last couple of years, i.e. the proliferation of the OS licenses. At the time of writing this paper, more than 70 different OS licenses have been registered; these licenses differ along several dimensions.[7] Take for instance the reciprocal provision: not all the OS licenses impose such provision,[8] while, at the same time, an extreme heterogeneity in terms of the degree of reciprocity imposed on derivative works can be observed between those licenses that do impose reciprocity. Interestingly, various commercial vendors have created their own open source license, thus suggesting a possible strategic role in the "design" of the license. This is the case for instance of IBM, Intel, Microsoft, or Nokia.[9]

---

[6]This is the case of, for instance, the so-called "patent termination clauses"; for a discussion of this issue see Rosen (2004).

[7]License proliferation is a very debated issue in the OS world; indeed, the presence of different schemes may pose serious problems given that some licenses are potentially incompatible with each other; for a discussion see Rosen (2004), chapter 10, and the report of the "License Proliferation Committee", available at the Open Source Initiative web-site, `www.opensource.org`.

[8]The most notable examples are represented by the BSD (Berkeley software distribution) and by the other so-called "academic licenses" that do not impose reciprocity.

[9]The case of Nokia is emblematic. At the url `opensource.nokia.com/` several different software projects are available for download and often projects are licensed under different terms. Some projects are distributed under the Nokia Open Source License, others are available under different OS licensing templates such as: GPL, BSD, Mozilla Public License, LGPL, and others.

The choice of releasing the source code to the OS community implies some risks too: although the copyright is still in the control of the software vendor that has originally developed the code,[10] the act of placing the software on a public repository stimulates competition. Customers that do not embed the software into their final products and/or that are not interested in other services usually bundled with the proprietary one, are unaffected by the restrictions coming with the OS version; obviously, these customers do not find significant differences (expect for the price) between the proprietary and the OS version of the software.[11] When these customers represent the vast majority of the market, the release of the code to the OS community comes at the serious risk of *cannibalization*.

In this paper we consider a profit maximizing firm that is developing a software project targeted to commercial customers. The project can be completely developed in-house or it can be released to the OS community in order to benefit from its contributions and feedbacks; in this latter case the firm employs a dual licensing strategy: it releases the code to the OS community, and it also sells a proprietary version. As we show in the paper, a crucial decision to be taken by the firm concerns the kind of OS license that regulates the distribution of the open source version of the software. The degree of reciprocity and the restrictions imposed on the OS usage affect both the extent to which the firm benefits from the contributions of the OS community, but they also represent a necessary safeguard against the risk of cannibalization.[12]

Our results suggest that the firm finds it optimal to employ a dual licensing strategy whenever the contribution from the OS community in terms of code enhancement is sufficiently large; the characteristics of the OS license need then to be set appropriately in order

[10]It is important to stress that by releasing the OS version of the software, a software firm does not give up the copyright on its code; indeed, at any moment in time, the firm can use the lines on which it has the copyright in order to offer a proprietary version of the software.

[11]Competition might arise in other forms too. OS programmers may download the release software and start independent development on it, thus creating a distinct and competitive piece of software that possibly overcomes in terms of quality or adoption the original one. In the industry jargon, this threat is known as *forking*.

[12]The software vendor can take other actions, not considered in our model, in order to protect its business. Adds-on, additional features and functionalities bundled into the proprietary version of the code represent a possible safeguard against cannibalization. Note that this behavior is not always effective since it might go against the "ethical" principles of the OS community which might feel exploited being released a downgraded version of the software code (Capobianco, 2006).

to reduce the competitive threat placed by the OS version. In this sense, dual licensing is a form of versioning,[13] where the software vendor releases the code to the OS community to improve its quality, and then it induces enough customers to buy the costly proprietary version through a proper "micromanagement" of the OS licensing terms.

We show that the characteristics of the OS license chosen by the firm depend on various market's attitudes towards reciprocity/restrictiveness; this is an interesting result which suggests a possible new explanation for the observed proliferation of OS licenses.

The model is presented in Section 2 and Section 3 concludes.

2 The model

Suppose that a commercial firm has started developing a new software which is directed to other commercial firms that need the source code of the software to embed it into their own products. These customers are interested not only in the quality of the code "per-se", but also to the terms of licensing under which it is distributed: since commercial customers want to keep full control on their products and/or are willing to receive fully featured products, they prefer to obtain the code under unrestrictive licenses and or bundled with a full provision of complementary services. For the sake o simplicity, we group all the possible dimensions that limit the use of the OS distribution within a single category that we define as OS license restrictiveness.

The firm may follow two alternative strategies to develop the software: it can either complete the project on its own, or it can release the code to the open source community in order to benefit from the contributions of the OS community, thus enjoying of the so called "development externality".[14] Clearly, in this latter case, the decision to go open source must be taken cautiously since it implies getting rid of the monopoly position.

The firm takes its decisions sequentially as follows:

[13] For details on the description and implementation of versioning strategies, see Shapiro and Varian (1998) and Bhargava and Choudhary (2001).

[14] Our model does not consider standard network effects based on usage; this type of externality is not relevant in our framework since at the equilibrium, also in case the software is kept proprietary, the market is fully covered and, irrespectively of the strategy adopted by the firm, the installed base of users is always equal to 1. Furthermore, as shown in Bonaccorsi and Rossi (2004) in a survey based on a sample of 146 Italian software firms, the development externality is the main motivation for going open source.

1. the firm decides whether to release the source code to the open source community; if it releases the code, the firm sets the degree of restrictiveness of the OS licence: formally, it chooses $r \in [0, \infty)$;

2. once the code has been developed, the firm chooses the price p of the proprietary version.

Customers observe the firm's licensing strategies and take their adoption decision.[15] For the sake of simplicity, all through the paper we normalize the firm's costs to zero and we assume that customers have mass 1.

Consumers may either adopt the proprietary or, whenever available, the OS version of the software; in the former case, a consumer obtains a net benefit equal to

$$U_P(p) = V + \theta N - p,$$

where $V + \theta N$ is the overall quality of the software, with V representing the contribution of the firm and θN the development externality accruing from the OS community, which is increasing in N, the mass of open source adopters, and in the strength of the externality, $\theta > 0$. Alternatively, whenever the firm has dual-lincensed its product, a customer can obtain the open source version of the code without paying any license fee; given r, the level of license restrictiveness set by the firm, the net benefit enjoyed upon the adoption of the OS version, is

$$U_{OS}(t, r) = V + \theta N - tr,$$

where $t\,r$ represents the disutility due to the restrictions imposed by the license.

We assume that customers have heterogeneous preferences with respect to license restrictiveness and we capture this feature through the parameter t, with $t \sim U(\mu - \delta, \mu + \delta)$ and where $\delta > 0$ and $\mu - \delta > 0$.[16]

[15] Note that according to the timing of the model some users may adopt the OS version at stage 1 and then contribute to the development of the code, while others will postpone their adoption decision after the proprietary version has been released. Assuming that customers *i*) do not derive additional benefits from adopting the OS version at stage 1 and *ii*) they rationally forecast the size of the OS community, then the exact timing of adoption decision is not relevant. Allowing customers to derive additional benefits from early adoption would not significantly change our results. This way of modeling the timing of adoption decision widely accepted in the literature; see Katz and Shapiro (1986).

[16] By definition, the proprietary version does not require any reciprocal provision, which is equivalent to having a software distributed under an unrestrictive licence, $r = 0$.

The disutility incurred by a customer when he/she adopts the OS version depends on the use and on the nature of the software. Since customers use the code as an input to produce other, derived, software that they either sell directly or that they embed into their own products, then t is larger when the derived software represents the core of the customers' products/technologies: the more relevant the derived software in the embedded system, the larger the damage for the embedder if forced to release the code under reciprocal licensing terms.

2.1 Results

In order to define the profits of the firm and to solve for its optimal strategy, we need to start by defining the size of the OS community, N. A consumer located at t adopts the OS version rather then the proprietary one when i) he/she enjoys a positive utility from OS adoption and ii) when this utility is larger than the one received by adopting the proprietary version; formally:

$$i)\ U_{OS}\left(t,r\right) \geq 0, \quad \Rightarrow \quad t \leq \frac{V+\theta N}{r},$$

and

$$ii)\ U_{OS}\left(t,r\right) \geq U_{P}\left(p\right) \quad \Rightarrow \quad t \leq \frac{p}{r}.$$

Using these two conditions, the size of the community of OS adopters can be defined as follows:

$$N = \int_{\mu-\delta}^{\min\left\{\frac{V+\theta N}{r},\frac{p}{r},\mu+\delta\right\}} \frac{1}{2\delta} dt, \tag{1}$$

with $N \geq 0$ provided that $\min\left\{\frac{V+\theta N}{r},\frac{p}{r},\mu+\delta\right\} \geq \mu-\delta$.

The firm decides the level of license restrictiveness and the price of the proprietary version to maximize its profits;[17] in order to sell to a strictly positive amount of users, r and p must be set such that a) individuals obtain a positive net benefit from buying the proprietary version and b) this benefit is larger than that enjoyed by adopting the OS version; formally:

$$a)\ U_{P}\left(p\right) \geq 0 \quad \Rightarrow \quad p \leq V+\theta N,$$

[17] Formally, the initial decision to be taken at stage 1 is whether to keep the software proprietary or to release the code also to the OS community. In our framework the former choice coincides with setting a sufficiently large r.

and

$$b)\ U_P(p) \geq U_{OS}(t = \mu + \delta, r) \quad \Rightarrow \quad p \leq r(\mu + \delta).$$

Inequality $b)$ ensures that at least the individual located at the extreme right of the segment representing the preferences towards license restrictiveness, namely the one that suffers the most from a given OS license r, must prefer the proprietary to the OS version; in this case, and provided that condition $a)$ applies, the firm gets a strictly positive profit. Conditions $a)$ and $b)$ imply that $\min\left\{\frac{V+\theta N(r,p)}{r}, \frac{p}{r}, \mu+\delta\right\} = \frac{p}{r}$;[18] therefore, from expression (1), the size of the OS community as a function of r and p simply reduces to:

$$N(r,p) = \begin{cases} \frac{\frac{p}{r}-(\mu-\delta)}{2\delta} & \text{if } p \leq V + \theta N(r,p),\, p \leq r(\mu+\delta),\ \text{and } p \geq r(\mu-\delta), \\ 0 & \text{if } p < r(\mu-\delta) \ \text{ or } r > \frac{V}{\mu-\delta}. \end{cases} \tag{2}$$

Note that no one adopt the OS version if either the proprietary software is sufficiently cheap or the former is released under a sufficiently restrictive license. The firm sets r, and afterwards p, in order to maximize its profits defined as:

$$\pi(r,p) = p\,[1 - N(r,p)], \tag{3}$$

where $N(r,p)$ is the expression defined in (2). We solve this maximization problem in the technical appendix; the solution highlights the firm's optimal strategy:

Proposition 1. *When $V < \theta$ the firm optimally employs a dual licensing strategy, with $r^* = \frac{\theta(V+\theta)}{2(\theta\mu-\delta V)}$, while it releases only the proprietary version otherwise.*

Proposition 1 highlights a clear result: the firm benefits from selecting a dual licensing strategy only when the strength of the development externality that it enjoys in this case is large relative to V. When θ takes smaller values and or the quality of the code produced by the firm V is already sufficiently large, then the firm optimally keeps the software proprietary and charges the monopoly price V.

It is interesting to note that the choice of endorsing a dual licensing strategy is entirely driven by a comparison between θ and V while it does not depend on the distribution of the individual preferences with respect to r. This observation is interesting although it is not surprising: it is by selecting an appropriate level of license restrictiveness, that the firm adapts its OS strategy to the specific characteristics of the customers' preferences. The following corollary describes in details the firm's optimal choice of r:

[18] Note also that the condition $\frac{p}{r} \leq \mu + \delta$ implies that $N(r,p) \leq 1$.

Corollary 1. *When the firm dual licenses, the optimal level of license restrictiveness* r^* *is such that:* $\frac{dr^*}{dV} > 0$*,* $\frac{dr^*}{d\delta} > 0$*,* $\frac{dr^*}{d\mu} < 0$*, and* $\frac{dr^*}{d\theta} < 0$ *when* $\mu > 3\delta$ *or when* $\mu < 3\delta$ *and* $\theta \in \left(0, V\frac{\delta+\sqrt{\delta(\mu+\delta)}}{\mu}\right)$*, while* $\frac{dr^*}{d\theta} > 0$ *otherwise.*

The positive relationship between r^* and V can be explained following the same arguments used to discuss Proposition 1: as V increases, the firm benefits from employing a more "proprietary strategy", i.e. by setting a more restrictive license in order to protect a valuable business.

Consider now the role of the customers preferences towards license restrictiveness, here represented by the parameters δ and μ. δ represents the dispersion of the distribution of customers' disutility towards r and according to Corollary 1, the firm tends to set a more restrictive license as δ increases. This result is driven by the trade-off faced by the firm: while, on the one side the firm wants to enjoy a sufficiently large development externality by stimulating OS adoption, on the other it needs to protect its business. When individuals are highly dispersed, there are customers with a very small t, i.e. which are willing to adopt the OS version even if the OS license imposes strong obligations. In this case, the firm can accomplish its first task of stimulating enough OS adoption, even by setting a relatively restrictive license. The effect of an increase in μ, the expected value of t, goes exactly in the opposite direction; as μ increases, the firm needs to reduce the restrictiveness of the OS license in order to benefit enough from the feedbacks and from the contributions of the open source community.

The impact of an increase in the strength of the development externality on r^* is more articulated and it entails to two opposite effects. A larger value of θ signals a more active OS community which provides a more valuable contribution to the development of the software; nonetheless, a larger θ makes the open source version of the software also a stronger competitor vis a vis the proprietary one. The former effect dominates whenever the size of the OS community is relatively small, that is when either the expected value of t is large relative to its dispersion (formally when $\mu > 3\delta$), or when the externality is not too strong. In both these cases, the firm benefits from enlarging the community through a reduction in the level of license restrictiveness. On the opposite, the "competition effect" prevails when the OS community is already sufficiently large; in this case, the firm reacts to a further increase in θ by selecting a larger r^*.

This simple model shows that dual licensing is an optimal strategy in a wide range of possible scenarios, thus explaining its growing popularity among software developers (Proposition 1). Furthermore, its prescriptions about the design of the OS license offer an interesting explanation to the observed proliferation of the OS licenses (Corollary 1). As discussed in Rosen (2004), in open source licenses there are many conditions other than reciprocity obligations that may be unacceptable to prospective licensees, such as the patent termination clauses that they incorporate, or the lack of warranties or indemnification. Our result suggests that the software firm can successfully dual license its product through the design of an appropriate and very specific license scheme, whose obligations depend on the characteristics of its customers.

3 Concluding remarks

An increasingly larger number of commercial software vendors is getting involved into the OS world. In this paper we propose a theoretical model to study the characteristics and the commercial sustainability of a particular open source strategy known as dual licensing. With dual licensing a firm offers one core software product under two different licensing schemes: a traditional proprietary license and an open source one.

In the paper we focus on the decision of a software vendor of whether to develop a fully proprietary version of the software or to employ a dual licensing strategy. Interestingly, we show that this decision is solely determined by the strength and the relevance of the contribution of the OS community: dual licensing is the optimal strategy whenever such contribution is sufficiently valuable.

This result points to the crucial role of OS licensing schemes for firms embracing open source strategies, and can help explaining the observed proliferation of the open source licenses. With an appropriate definition of the licensing terms under which the OS version of the software is distributed the firm can balance the opposing effects of going open source, namely it can trade-off the benefits obtained from the contributions of the OS community, and, the risk of cannibalizing its business.

Even though the paper focuses on dual licensing and on the role of the reciprocal provision the intuition of our results is broader and may be applied to more general open source strategies. As discussed in the introduction, different dimensions of OS licenses may be undesirable for some customers; in this cases by an appropriate definition of the licensing terms, the software house can benefit from the contribution of the OS community and sell an upgraded version of the software (either proprietary or still open source) to those customers who are willing to pay to get rid to some specific provision included into the free version.

Technical appendix

Proof of Proposition 1. We proceed by backward induction; at the second stage, the firm optimally choses the price, for any r set at stage 1. Expression (2) shows the three constraints that ensure an interior solution; looking at these constraints, it is useful to define the following expressions: $(1) \equiv r(\mu - \delta)$, $(2) \equiv r(\mu + \delta)$ and $(3) \equiv \frac{2\delta V - \theta(\mu-\delta)}{2\delta r - \theta} r$. The three constraints in expression (2) can be simply rewritten as follows: $p \geq (1)$, $p \leq (2)$ and, finally, $p \geq (3)$ when $r < \frac{\theta}{2\delta}$, and $p \leq (3)$ otherwise.

Simple algebraic inspection of expressions (1), (2) and (3) reveals, for any given r, which are the relevant constraints. Clearly, it is always true that $(1) < (2)$.

Furthermore, by contrasting (3) with (1) and (2), it follows that: *i*) when $r > \frac{\theta}{2\delta}$, then $(3) \geq (1)$ iff $r \leq \frac{V}{\mu-\delta}$ and $(3) \geq (2)$ iff $r \leq \frac{V+\theta}{\mu+\delta}$, and *ii*) when $r < \frac{\theta}{2\delta}$, then $(3) \geq (1)$ iff $r \geq \frac{V}{\mu-\delta}$ and $(3) \geq (2)$ iff $r \leq \frac{V+\theta}{\mu+\delta}$.

It is possible to show that the various thresholds can be ordered in the following way: $\frac{\theta}{2\delta} \geq \frac{V+\theta}{\mu+\delta} \geq \frac{V}{\mu-\delta}$ when $V \leq \frac{\theta(\mu-\delta)}{2\delta}$, while $\frac{\theta}{2\delta} < \frac{V+\theta}{\mu+\delta} < \frac{V}{\mu-\delta}$ otherwise.

These orderings allow us to partition the parameters' space in two regions; for the moment let us focus on the first region, i.e. when $V \leq \frac{\theta(\mu-\delta)}{2\delta}$. In this case, it is possible to verify that the constraint $p\left(\frac{2\delta r-\theta}{r}\right) \leq 2\delta V - \theta(\mu - \delta)$ cannot be satisfied when $r \geq \frac{\theta}{2\delta}$ since it would require $p \leq \frac{2\delta V-\theta(\mu-\delta)}{2\delta r-\theta} r < 0$. Let us first focus to the case where $r < \frac{\theta}{2\delta}$; the firm's second stage maximization problem reduces to:

$$\max_p \pi(r,p) = p\left(1 - \frac{\frac{p}{r} - (\mu - \delta)}{2\delta}\right) \tag{4}$$
$$\text{s.t. } p \geq (1), p \leq (2), p \geq (3)$$

For future reference, it is useful to derive the price that would maximize the unconstrained profit function; taking the derivative of $\pi(r,p)$ with respect to p, the unconstrained optimum is: $\frac{(\mu+\delta)}{2} r$.[19]

We need to define the optimal price schedule $p^*(r)$ that for any given level of r solves the maximization problem (4); according to the first ordering presented above, three possible cases must be considered: *a*) $r < \frac{V}{\mu-\delta}$, *b*) $\frac{V}{\mu-\delta} \leq r < \frac{V+\theta}{\mu+\delta}$, and *c*) $r \geq \frac{V+\theta}{\mu+\delta}$ (provided that $r < \frac{\theta}{2\delta}$).

Case *a*): $r < \frac{V}{\mu-\delta}$. In this case $(3) < (1)$, therefore the relevant constraints are $p \geq r(\mu - \delta)$ and $p \leq r(\mu + \delta)$. It can be easily shown that the optimal price set by the firm in the second stage is:

$$p^*(r) = \begin{cases} \frac{(\mu+\delta)}{2} r & \text{if } \mu \leq 3\delta, \\ r(\mu - \delta) & \text{otherwise.} \end{cases}$$

[19] It is easy to verify that the second order condition is satisfied.

Plugging this price schedule into the firm's profits, it follows that the first stage profit function is:

$$\pi^*(r) = \begin{cases} \frac{r(\mu+\delta)^2}{8\delta} & \text{if } \mu \leq 3\delta, \\ r(\mu-\delta) & \text{otherwise.} \end{cases}$$

Consider now case b): $r \in \left[\frac{V}{\mu-\delta}, \frac{V+\theta}{\mu+\delta}\right)$; in this case $(1) \leq (3)$, and the relevant constraints are $p \geq \frac{2\delta V-\theta(\mu-\delta)}{2\delta r-\theta} r$ and $p \leq r(\mu+\delta)$. The optimal price for this region of the parameters is:

$$p^*(r) = \begin{cases} \frac{(\mu+\delta)}{2} r & \text{if } \mu \leq 3\delta \text{ and } r \in \left[\frac{V}{\mu-\delta}\tilde{r}\right), \\ \frac{2\delta V-\theta(\mu-\delta)}{2\delta r-\theta} r & \text{if } \mu \leq 3\delta \text{ and } r \in \left[\tilde{r}, \frac{V+\theta}{\mu+\delta}\right), \\ \frac{2\delta V-\theta(\mu-\delta)}{2\delta r-\theta} r & \text{if } \mu > 3\delta. \end{cases}$$

where $\tilde{r} = \frac{2\delta(2V+\theta)-\theta(\mu-\delta)}{2\delta(\mu+\delta)}$.[20] Plugging this price schedule into $\pi(r,p)$, it follows that:

$$\pi^*(r) = \begin{cases} \frac{r(\mu+\delta)^2}{8\delta} & \text{if } \mu \leq 3\delta \text{ and } r \in \left[\frac{V}{\mu-\delta}\tilde{r}\right) \\ \frac{(2\delta V+\theta\delta-\theta\mu)(\delta r-\theta-V+\mu r)}{(2\delta r-\theta)^2} r & \text{if } \mu \leq 3\delta \text{ and } r \in \left[\tilde{r}, \frac{V+\theta}{\mu+\delta}\right) \\ \frac{(2\delta V+\theta\delta-\theta\mu)(\delta r-\theta-V+\mu r)}{(2\delta r-\theta)^2} r & \text{if } \mu > 3\delta \end{cases}$$

Finally, we are left with case c): $r \geq \frac{V+\theta}{\mu+\delta}$. In this case $(3) \geq (2)$; it can be verified that in this region of the parameters' space, the three constraints of the maximization problem cannot be simultaneously satisfied. Therefore, the firm sells the proprietary version to all customers at $p = V$ and gets profits equal to V; this result extends also to the case where $r \geq \frac{\theta}{2\delta}$.
We are now ready to move back to the first stage of the game and to look for the optimal license restrictiveness. From the above analysis two relevant cases can be envisaged: I) $\mu \leq 3\delta$, and II) $\mu > 3\delta$.
Case I). When $\mu \leq 3\delta$ the profit function is:

$$\pi^*(r) = \begin{cases} \frac{r(\mu+\delta)^2}{8\delta} & \text{if } r \in [0, \tilde{r}) \\ \frac{(2\delta V+\theta\delta-\theta\mu)(\delta r-\theta-V+\mu r)}{(2\delta r-\theta)^2} r & \text{if } r \in \left[\tilde{r}, \frac{V+\theta}{\mu+\delta}\right) \\ V & \text{otherwise} \end{cases}$$

$\pi^*(r)$ is increasing in r for $r \in [0, \tilde{r})$, while it does not depend on r for $r \geq \frac{V+\theta}{\mu+\delta}$; let us consider the intermediate values of r. The derivative of the profit function in case $r \in \left[\tilde{r}, \frac{V+\theta}{\mu+\delta}\right)$ is:

$$\frac{d\pi^*(r)}{dr} = (-2\delta V + \theta\mu - \theta\delta) \frac{2r(\mu\theta - \delta V) - \theta(V+\theta)}{(2\delta r - \theta)^3}.$$

The solution of the first order condition is $r^* = \frac{\theta(V+\theta)}{2(\theta\mu-\delta V)}$.[21] Simple algebra shows that $r^* \in \left(\tilde{r}, \frac{V+\theta}{\mu+\delta}\right)$ provided that $V < \frac{\theta(\mu-\delta)}{2\delta}$; furthermore, it is relatively easy to verify that $\pi^*(r)$ is continuous at

[20]Note that the condition $V < \frac{\theta(\mu-\delta)r}{2\delta}$ ensures that $\tilde{r} \in \left(\frac{V}{\mu-\delta}, \frac{V+\theta}{\mu+\delta}\right)$.
[21]The second order condition at $r = r^*$ is satisfied.

$r = \tilde{r}$, and that $\pi^* (r = r^*) > V$. This is enough to prove that for $V \leq \frac{\theta(\mu-\delta)}{2\delta}$ and $\mu \leq 3\delta$ it is optimal for the firm to employ a dual licensing strategy and release the code to the OS community at $r = r^*$.

Case II): $\mu > 3\delta$; in this case, firm's profits are:

$$\pi^* (r) = \begin{cases} r (\mu - \delta) & \text{if } r \in \left[0, \frac{V}{\mu-\delta}\right), \\ \frac{(2\delta V + \theta\delta - \theta\mu)(\delta r - \theta - V + \mu r)}{(2\delta r - \theta)^2} r & \text{if } r \in \left[\frac{V}{\mu-\delta}, \frac{V+\theta}{\mu+\delta}\right), \\ V & \text{otherwise.} \end{cases} \tag{5}$$

Similarly to the previous case, $\pi^* (r)$ is increasing in r for $r \in \left[0, \frac{V}{\mu-\delta}\right)$, while it is constant at V for $r \geq \frac{V+\theta}{\mu+\delta}$. Exactly as above, for intermediate values of r, formally for $r \in \left[\frac{V}{\mu-\delta}, \frac{V+\theta}{\mu+\delta}\right)$, $\pi^* (r)$ has a maximum at r^*. We already know from the previous analysis that $r^* < \frac{V+\theta}{\mu+\delta}$; moreover, it is possible to check that $r^* \geq \frac{V}{\mu-\delta}$ for $V < \theta$ and $r^* < \frac{V}{\mu-\delta}$ for $V \in \left[\theta, \frac{\theta(\mu-\delta)}{2\delta}\right)$. Therefore the optimal licence is $r = r^*$ for $V < \theta$ and $r = \frac{V}{\mu-\delta}$ for $V \in \left[\theta, \frac{\theta(\mu-\delta)}{2\delta}\right)$. One can check that for $r = \frac{V}{\mu-\delta}$, the piecewise profit function described in (5), collapses to V; formally, for $r = \frac{V}{\mu-\delta}$, $r (\mu - \delta) = \frac{(2\delta V + \theta\delta - \theta\mu)(\delta r - \theta - V + \mu r)}{(2\delta r - \theta)^2} r = V$.

Combining the results obtained in case I) and II), it emerges that the optimal strategy for the firm is the following: when $V < \theta$, the firm maximizes its profit by employing a dual licensing strategy with $r = r^*$, while when $V \in \left[\theta, \frac{\theta(\mu-\delta)}{2\delta}\right)$ it is optimal to set $r = \frac{V}{\mu-\delta}$ or any $r \geq \frac{V+\theta}{\mu+\delta}$. It is possible to check that in both these latter cases the size of the OS community is zero, which implicitly means that the firm keeps the software proprietary.

In order to complete the proof we should consider the case $V > \frac{\theta(\mu-\delta)}{2\delta}$. Proceeding in the same way as above, it can be shown that the optimal strategy for the firm is to release only the proprietary version of the software when $\mu > 3\delta$; when $\mu \leq 3\delta$, it is optimal to dual license at r^* if $V \in \left(\frac{\theta(\mu-\delta)}{2\delta}, \theta\right)$ and release only the proprietary version otherwise.[22] Putting all these various cases together the proposition follows suit. □

[22] The algebra for the case $V > \frac{\theta(\mu-\delta)}{2\delta}$ omitted for sake of brevity and it is available upon request from the authors.

References

Bhargava, H. and Choudhary, V. (2001). Second Degree Price Discrimination for Information Goods under Nonlinear Utility Functions. In *HICSS '01: Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 7*, page 7039, Washington, DC, USA. IEEE Computer Society.

Bonaccorsi, A., Giannangeli, S., and Rossi, C. (2006). Entry Strategies under Competing Standards: Hybrid Business Models in the Open Source Software Industry. *Management Science*, 52(7):1085–98.

Bonaccorsi, A. and Rossi, C. (2004). Altruistich individuals selfish firms? The structure of motivation in open source software. *First Monday*, 9(1).

Capobianco, F. (2006). My Honest Dual Licensing. Unpublished manuscript, available at www.funambol.com/blog/capo/2006/07/my-honest-dual-licensing.html.

Hecker, F. (1999). Setting Up Shop: the Business of Open-Source Software. *IEEE Software*, 16(1):45–51.

Katz, M. and Shapiro, C. (1986). Technology Adoption in the Presence of Network Externalities. *Journal of Political Economy*, 94(4):822–841.

Koski, H. (2007). Private-collective Software Business Models: Cordinatitons and Commercialization via Licensing. Discussion Papers 1091, The Research Institute of the Finnish Economy. Available at http://ideas.repec.org/p/rif/dpaper/1091.html.

Moglen, E. (2003). Free Software Foundation Statement on SCO v. IBM. Document retrieved from www.fsf.org/licensing/sco/sco-v-ibm.html.

Olson, M. (2006). *Dual Licensing.* In DiBona C., Canese C. and Stone M.: Open Sources 2.0: The Continuing Evolution, O'Reilly Media.

Rajala, R., Nissilä, J., and Westerlund, M. (2007). *Revenue Models in the Open Source Software Business.* In Kirk St. Amant and Brian Still eds. Handbook of Research on Open Source Software: Technological, Economic, and Social Perspectives, IGI Global.

Rosen, L. (2004). *Open Source Licensing. Software Freedom and Intellectual Property Law.* Prentice Hall.

Shapiro, C. and Varian, H. (1998). Versioning: The Smart Way to Sell Information. *Harvard Business Review*, 76(6):106–114.

Välimäki, M. (2005). *The Rise of Open Source Licensing: A Challenge to the Use of Intellectual Property in the Software Industry.* Turre Publishing.

West, J. and Gallagher, S. (2004). Key Challenges of Open Innovation: Lessons from Open Source Software. San José State University, mimeo; document avaiable at www.joelwest.org/Papers/WestGallagher2004.pdf.