Routledge
Taylor & Francis Group

Economics of Innovation and New Technology

ISSN: 1043-8599 (Print) 1476-8364 (Online) Journal homepage: https://www.tandfonline.com/loi/gein20

# On substitution of intellectual property and free disclosure: an analysis of R&D strategies in software technologies

Elad Harison & Robin Cowan

**To cite this article:** Elad Harison & Robin Cowan (2004) On substitution of intellectual property and free disclosure: an analysis of R&D strategies in software technologies , Economics of Innovation and New Technology, 13:5, 477-487, DOI: 10.1080/1043859042000253581

**To link to this article:** https://doi.org/10.1080/1043859042000253581

*Econ. Innov. New Techn.,* 2004, Vol. 13(5), July, pp. 477–487

Taylor & Francis
Taylor & Francis Group

# ON SUBSTITUTION OF INTELLECTUAL PROPERTY AND FREE DISCLOSURE: AN ANALYSIS OF R&D STRATEGIES IN SOFTWARE TECHNOLOGIES

ELAD HARISON* and ROBIN COWAN†

*MERIT, University of Maastricht, PO Box 616, 6200 MD Maastricht, The Netherlands*

*(Received 17 September 2002; In final form 11 November 2003)*

Major firms have joined the open-source movement and have chosen to apply that development methodology in their projects. Our model examines the links between openness and innovation in software technologies by revealing how disclosure affects the technical quality of computer applications and the profits of myopic and far-sighted firms. The model analyzes the degree of disclosure that should be implemented to optimize profits in various market scenarios. Further, we reveal how social welfare of users (in terms of technical quality of the products that they implement) relates to profit-maximization decisions of the firm. If revenue is unresponsive to openness or slowly responds to it, the firm would prefer to leave the source code proprietary. Otherwise, if the market conditions change and the effective revenue increases rapidly enough with openness, the optimal strategy changes from entirely proprietary to some open-source development.

*Keywords*: Intellectual property; Software; Open source; Disclosure; Technical quality

*JEL classification*: L15; L86; O34

## 1 INTRODUCTION

Since the beginning of the 1990s, we have seen a significant increase in the development of computer programs as open-source applications, publicly offered with a free-use and distribution license. Prominent and widely implemented examples are *Linux*, *Apache*, and *Sendmail*. Part of the success of the open-source movement is explained by the rapid growth of the internet as a communication network between users and developers. In turn, this permits the community to turn widely distributed programing efforts and skills into a continuous improvement of computer programs and solid technical support at virtually no cost (Lakhani and von Hippel, 2000). While the early modern history of computer software involved considerable open-source development, in recent decades, particularly since the rise of the personal computer, open source had moved to the fringe of the industry. However, in the past few years major firms have decided to integrate open-source methodologies in part of their projects. Despite an extensive legal framework, which permits software developers to appropriate rents from their programs and

* Corresponding author. Tel.: 00-31-43-3883883; Fax: 00-31-43-3884905; E-mail: e.harison@merit.unimaas.nl;
† E-mail: r.cowan@merit.unimaas.nl

ISSN 1043-8599 print; ISSN 1476-8364 online 
DOI:10.1080/1043859042000253581

inventions,[1] many developers prefer to offer their creative output at zero revenue. Firms that supply open source as part of their business model expect to raise their profits and to increase their market shares by disclosing their core asset for free-use, modification, and re-sale by removing any intellectual property claims from it.[2] Alternatively, a firm's decision to embrace open source involves the expectation that by disclosing contents of their technology to the public domain, complementary services (e.g. implementation and technical support) can be exploited commercially and will expand through a rapid diffusion of freely offered contents.[3] As the use of software applications expands, so do their 'production externalities': Knowledge spillovers that foster technological advance increase with the level of openness of the source code, as skilled users integrate new features into the base program. Therefore, when increasing the numbers of subroutines that are made public, the technical quality and the performance of the product is more likely to rise.[4] Further, by 'outsourcing' the tasks of R&D to open-source communities, software firms are able to reduce their development costs.[5] However, the major drawback in a firm's decision to adopt open-source methodologies is the removal of intellectual property claims from disclosed features, as a precondition for involving external programers in the development process. Consequently, the revenues from open versions may become lower than revenues from proprietary versions. Hence, many software firms choose an alternative strategy and distribute their products as *hybrids*, in which part of the source code remains proprietary and the other is made open by disclosing their code-lines or by customizing the licensing terms of their products (McKelvey, 200l).[6]

The development and the distribution strategy of firms closely affect the licensing terms of software. These, in turn, influence the conditions and restrictions for future 'generations' and releases of the technology and its derivative and complementary applications. To clarify, in the model that follows we assume that open-source programs are distributed under permissive terms of 'X-license' that enable free-use, commercialization, development of new features, and application of a 'mixture' of free, open, and proprietary software. By using the model as a baseline case, future research may assess the impact of the various licenses present in the open and free software community (e.g. GPL and BSD) on the development of information technologies. However, in some cases analytical modeling can be far more complicated than the general model presented here, considering the 'viral' characteristics of licensing terms and the opposing views on the links between Copyleft licenses and their contribution or impediment to innovation (contrast, for example, Reese and Stenberg, 2001 with Lessig, 2002; Smith, 2002).

[1] Increasing numbers of software patents submitted and approved by the United States Patent and Trademark Office and the rising number of lawsuits over patent disputes in information technologies reflect a wide adoption of IPRs as strategic means by software companies (Granstrand, 2000; Cowan and Harison, 2001).

[2] A recent example is the release of the source code of IBM's Eclipse project, a new software platform on which information systems are built, in November 2001. It was a proprietary software that was developed at the cost of forty million dollars (CNET News, Nov. 2001). Other firms, in particular small and medium enterprises, apply open source to gain a competitive advantage by expanding the installed base of users of their technologies.

[3] *Red Hat Linux* is a prevalent example for a successful business model fully based on open-source software. The company distributes open-source applications, which could be downloaded for free from the internet, but Red Hat provides its customers full guarantee and technical support. Red Hat acquires the source code at no cost, tests and improves the software, and then sells it in the market. Although Red Hat Linux may be installed and used for free (both are permissible by Linux licensing terms), most of its customers prefer to buy an original copy of the software, as an 'insurance premium', and enjoy the firm's guarantee (Young, 1999).

[4] Results of several benchmark studies comparing the main variables of performance and stability in open-source versus. proprietary applications have shown superiority of the open-source Linux over its proprietary rival, Windows 2000 (see for instance: Rothman and Buckman, 2001; PC Magazine, 2001, 2002).

[5] Software firms had long ago reduced their testing costs by introducing *beta versions* of computer applications to potential users before releasing the final products to the market.

[6] For example, the Sun Community Source Licensing for its Java-based applications integrates both open source and proprietary policies according to the use of those products. Users can download the Java developers' kits and use it at no cost. However, licensing fees are charged when they acquire the source code of the standard environment and modify it for their own purposes.

A positive effect of free dissemination of software products, which even exists in software piracy, is the increasing number of users that choose to adopt them (i.e. the *installed base* of computer programs). Firms may benefit from disseminating unprotected versions since the demand and the propensity to pay higher revenues for legally purchased copies increase with the total number of licensed *and* unlicensed users of their products (Conner and Rumelt, 1991; Shy and Thisse, 1999). However, this strategy would succeed only if the elasticity of substitution between paid and unpaid use is larger than 1, i.e. the growth in profits from purchased copies exceeds the losses from illegal duplication. Yet, research on the economic effects of software piracy and unauthorized use examines software programs mainly as final goods (i.e. compiled software packages), overlooking one type of benefit that is involved in the disclosure of source code. Open-source projects are not necessarily recognized as successful by 'locking-in' the majority of consumers to a single-market standard over rival applications, but rather by attracting many agents who implement them, provide technical support to other users, and produce new and improved versions on a continuous basis.

On the supply side, various scholars suggest that developers participate in open-source projects and disclose their intellectual and professional output to the public at 'no cost' are driven by behavioral determinants, such as self-satisfaction or 'altruism', rather than by a more 'traditional' economic rationale.[7] Others argue that new opportunities to 'socialize' with members of online communities who share common interests and background and the reputation gained in professional circles are the main motives behind partaking open source (Lerner and Tirole, 2002). However, whether one social factor is more dominant than the others, common observations mention that the development of open sources is essentially not affected by any changes in the market and can be represented in economic terms as a non-elastic supply of skills.[8]

In the following sections, we look at the underlying dynamics of software markets, where producers of software platforms and major applications (such as operating systems or internet communication servers) are able to choose different degrees of disclosure, in terms of technical quality and profits. Our model analyzes how different degrees of source-code disclosure affect the performance of software products and technologies and the profitability of their producers. Section 2 describes a model of R&D in software firms where a short-sighted firm applies level of disclosure that optimizes single-period profits.

Section 3 expands the model and analyzes how disclosure influences long-term profits and social welfare and whether both measures correspond to each other. Section 4 expands the scope of the discussion and illuminates potential strategies in some common market scenarios. Finally, we compare between markets of myopic and forward-looking firms, where similar degrees of disclosure generate different levels of profitability and future investments in R&D and draw conclusions on the links between software disclosure, technical quality, and profits.

## 2 INTELLECTUAL PROPERTY VERSUS DISCLOSURE: A MODEL OF A MYOPIC FIRM

Software products and technologies are composed of different *technological features*, various components that execute information processing and computational functions, which are

[7] Although it is reasonable to assume that professional programers would prefer developing in their spare time commercial applications rather than free software, in any open-source community roughly half the developers are professional programmers (Hertel *et al*., 2003).

[8] The phenomenon of open-source communities is more likely to be found in information technologies rather than in electronic engineering or biotechnology, since new entrants are not required to make high capital investments in hardware in order to participate in 'production' of new computer programs.

involved in the operation of computer programs. Software features are commonly integrated to a single, coherent application and do not overlap either in their functionality or tasks.

The essence of the model is based on a firm's decision to disclose part of its source code to improve its profits. Since the disclosed features are freely disseminated and used, the firm must 'compensate' for its disclosure through increased revenues from the remaining proprietary features. Open-source features become available to both users and rival firms, thus we can assume that revenues are not generated here. Again this assumption is a way of simplifying the model. If the software market has a monopolistic competition structure, profits are made through product differentiation. But thinking in Lancasterian terms, if two products share a feature, there is little ability for the two firms to differentiate using that feature. Thus, the ability to differentiate, which is central to the ability to earn profits, depends on having features that other products or firms do not have. This is captured in a very straightforward way by assuming that firms earn no revenues on anything developed as open source (since competitors can easily copy them): their revenues must come from the features, which differentiate their products from the competitors. This approach implies that software is hybrid in that some parts of it are open and other parts are closed. Hybrid software may be relatively uncommon, but some important examples exist. The new Macintosh operating system is an example of a hybrid product: the heart of it is based on a BSD Unix version (called Darwin), on top of which is a proprietary user interface. The LindowsOS is another example for an operating system that was built on the basis of Debian Linux. The system offers compatibility to Windows applications (which does not exist in Linux versions), graphic, and user interfaces similar to WindowsXP and advanced features (e.g. anti-virus, anti-spam filters, and WiFi detector). At the same time, the model can be interpreted as representing a firm's portfolio of software. Some products are offered as open source, and others are proprietary. This avoids the issue of hybrid packages *per se*.

The model presents a stylized version of a real-world 'software consumption' behavior, in particular as regards the demand side. Since the links between revenue and openness are in principle unclear, we look instead for bounds on revenue response, such that open source becomes a viable development mode. While this makes the model appear simplified and quite unrealistic in a way, it captures the basic motivations of firms to develop their products as open or proprietary versions and avoids the difficulties involved in a fully blown demand side.

Following Nordhaus (1969, Ch. 2) model of technical change, the quality and the performance of technology expand when new features are added, either by the firm or by users. The incremental change in the *technical quality* of the software is defined as $R$.[9]

Assume for simplicity that development of new features involves no risk. Therefore, development of features accomplished by users substitutes for R&D of the firm, if the code-lines of this feature were disclosed.

The model addresses the problem of a single firm. Given the costs of the two modes of development, and the relationship between openness and revenue, the firm maximizes profits by choosing the degree of openness, $\alpha \in [0, 1]$ (the complementary share of the technology, $1 - \alpha$, remains proprietary and developed by the firm in-house).[10] Initially we examine the case in which revenues do not respond at all to openness, and then ask about conditions on the

[9] Technical quality signifies the performance of technology and can be described as a position on a 'quality ladder' (Grossman and Helpman, 1991). Following this definition, firms can continuously improve the technical quality of their products by investing in R&D.

[10] At one extreme lies full protection by IPRs and consequently full ownership of the technology. At the other extreme technology is developed only by public communities and, hence, can be freely copied, disseminated, and installed. In reality, however, full disclosure is more of a descriptive expression rather than an existing state, as part of the technical know-how always remains tacit within the domain of its inventors. The scenario of complete protection is far from reality as well as firms disclose essential software components to developers of complimentary applications and reveal their technical specifications. We assume that the technical know-how obtained from the share of proprietary software is only marginal.

relationship between revenue and openness that would encourage or discourage firms to open their source.

The cost to the firm of developing a new feature is $\bar{c}$ if it is developed in-house and $c_1$ if accomplished through open-source communities. Since open-source programers carry out major tasks, without compensation from the firm, $c_1 < \bar{c}$. Consequently, the R&D expenditure of the firm includes the cost of developing proprietary features, $\bar{c}(1-\alpha)R$, and the cost of open-source features, $c_1 \alpha R$.

The revenue of a single proprietary feature is a function of the level of disclosure, $\Psi(\alpha)$.[11] Revenues are generated only from features that are distributed as proprietary sources, $(1-\alpha)R$. One-period profits of the firm are achieved by subtracting its development costs on open source and proprietary features from its revenues.

The firm maximizes its one-period profits, $\pi$, by evaluating the trade-off between R&D expenditure on open-source features versus its expenditure on proprietary features and the revenues that are achieved:

$$\pi = (1-\alpha)\Psi(\alpha)R - [\bar{c}(1-\alpha)R + c_1 \alpha R] = (1-\alpha)\Psi(\alpha)R - [\bar{c} - \alpha(\bar{c} - c_1)]R. \quad (1)$$

Figure 1 illustrates the mechanism graphically, showing the consistency requirements among the variables. Given the change in the technical quality of software, $R$ (the horizontal axis in the north-east quadrant), a firm chooses a degree of openness $\alpha_1 \cdot \alpha_1$ is the argument in the revenue equation, $\Psi(\alpha)$, and the revenue is determined in the south-west, quadrant. The revenue $\Psi_1$ determines profits in the following way. Revenue is generated only on closed software, and is defined as $\Psi_1(1-\alpha_1)R$ (light gray plus medium gray). Total costs are the sum of open development costs (dark gray) plus closed costs (medium gray). Thus, profits are the difference

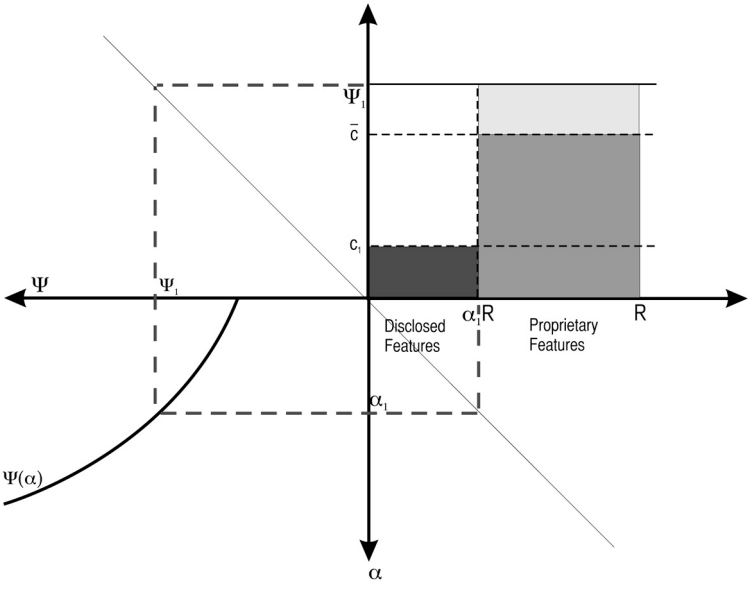


FIGURE 1 Illustration of the single-period model. For a given $R$, changes in $\alpha$ change both the proportion of $R$ on which the firm receives revenue, and, through $\Psi(\alpha)$, the revenue 'per feature' for those features.

[11] If a firm includes some proprietary and some open-source features in its product, rivals can duplicate the non-proprietary features in their own versions of the good. Thus, ignoring the possibility that users are interested in 'bundles' rather than features, a firm can only extract revenues on its proprietary features.

between profits on closed software and open development costs (light gray minus dark gray). Increasing $\alpha$ can increase the revenue per feature, but it will decrease the quantity of software on which the firm earns revenues. This captures the firm's basic trade-off.

A marginal increase in the degree of disclosure has three effects on the costs and gains of the firm: (1) Revenues decrease by $\Psi(\alpha)R - (1-\alpha)(\mathrm{d}\Psi(\alpha)/\mathrm{d}\alpha)R$ as the firm expands the non-remunerative segment of the product, (2) the total costs of open-source development increase by $c_1 \mathrm{d}\alpha R$, and (3) the costs of in-house development decrease by $\bar{c}\ \mathrm{d}\alpha R$.

Solution of the firm's problem depends heavily on revenue responsiveness. First, we examine the extreme case in which revenue does not respond to openness.

Differentiate Eq. (1) to find the cumulative effect on the profits of the firm:

$$\frac{\mathrm{d}\pi}{\mathrm{d}\alpha} = -\Psi(\alpha)R + (1-\alpha)\frac{\mathrm{d}\Psi(\alpha)}{\mathrm{d}\alpha}R + (\bar{c} - c_1)R. \tag{2}$$

PROPOSITION 1 *When the revenue per module is fixed* ($\Psi(\alpha) = \Psi$), *the optimal strategy is to keep all software features proprietary.*

From Eq. (2) $(\mathrm{d}\pi/\mathrm{d}\alpha) < 0$, so $\alpha^* = 0$.

If revenues do not respond to openness, the firm would prefer to keep its features undisclosed. However, an observation in today's software market is that firms are changing (notably, increasing) the degree of openness in their software development. In the extreme, they can extract no revenues from open-source software. Thus, a change in the degree of openness must be accompanied by a change in revenues of proprietary features to maintain the level of firm's profits. Following Eq. (2), we identify the conditions on revenue responsiveness for which (partial) openness is applied by the firm.

PROPOSITION 2 *If* $(\partial\Psi/\partial\alpha) > [(1/1-\alpha)(\Psi(\alpha) - (\bar{c} - c_1))]$ *at* $\alpha = 0$ *and* $\Psi(1) > (\bar{c} - c_1)$, *then* $0 < \alpha^* < 1$.

An interesting benchmark then is the relationship between $\alpha$ and $\pi$ such that profits do not change as $\alpha$ changes.

Differentiate Eq. (1) and set to zero. Rearranging:

$$\Psi(\alpha)\Big|_{\alpha=\alpha_1} = (1-\alpha)\left.\frac{\mathrm{d}\Psi(\alpha)}{\mathrm{d}\alpha}\right|_{\alpha=\alpha_1} + (\bar{c} - c_1), \tag{3}$$

or

$$\left.\frac{\mathrm{d}\Psi(\alpha)}{\Psi(\alpha) + (\bar{c} - c_1)}\right|_{\alpha=\alpha_1} = \left.\frac{\mathrm{d}\alpha}{1-\alpha}\right|_{\alpha=\alpha_1}. \tag{4}$$

By integrating the terms in Eq. (4), we obtain a revenue function for stable profits:

$$\Psi^*(\alpha) = (\bar{c} - c_1) + \frac{\Psi_0}{1-\alpha}, \tag{5}$$

where $\Psi_0$ is constant. By substituting Eq. (5) into Eq. (1), we can generate the non-negative profits constraint: $\Psi_0 > c_1$.

Following this line of discussion we can state the following proposition.

PROPOSITION 3 *If* $\Psi(\alpha)$ *is continuous and* $\Psi(\alpha^*) \gtrless (\bar{c} - c_1) + (c_1/1-\alpha)$, *then a firm operating with* $\alpha = \alpha^*$ *will increase* (*decrease*) *its level of openness.*

We can summarize these results briefly. If revenue is unresponsive to openness, software is only ever released as a proprietary product (Proposition 1). Further, if revenue responds slowly to openness, source code stays closed. However, if revenue is highly responsive to openness, the firm would open its source code. But since no revenue on fully open source is generated, part of the features remain proprietary (Proposition 2). When there is an exogenous change in the relationship between revenue and openness, Proposition 3 describes the conditions on the change such that a firm will increase or decrease its degree of openness.

## 3 FORWARD-LOOKING FIRMS

We extend the model to include intertemporal dynamics, keeping the basic terms and definitions of the previous model unchanged.[12] Define $R(t)$ as the increment of quality of software in period $t$ from the addition of new features either by the users or by the firm. The firm decides *ex ante* what share of features, $\alpha \in [0, 1]$, is developed as open source. The revenue received for a single proprietary feature changes with the degree of disclosure. In this dynamic setting of the model, we re-interpret $\Psi(\alpha)$ as the present discounted value of the gross revenue from a closed-source module or feature. Thus, $\Psi(\alpha(t))R(t)(1 - \alpha(t))$ is the present discounted value of the revenue from the technical advance in period $t$:

$$\mathrm{RV}(t) = (1 - \alpha)\Psi(\alpha)R(t). \tag{6}$$

The total expenditure of the firm on R&D in period $t$ is allocated between development of open and closed sources:

$$\mathrm{RD}(t) = \alpha R(t)c_1 + (1 - \alpha)R(t)\bar{c} = [\bar{c} - \alpha(\bar{c} - c_1)]R(t). \tag{7}$$

Assume that a constant share of firm's profits in each period, $\rho \in (0, 1)$, is invested in R&D in the following period.[13] Therefore, the total investments in R&D are determined by the profits of the firm in the preceding period, $\pi(t - 1)$, so $\mathrm{RD}(t)$ is:

$$\mathrm{RD}(t) \equiv \rho\pi(t - 1). \tag{8}$$

Subtract the cost of R&D from the revenues of the firm (Eq. (6)) to obtain the net present value (NPV) of features introduced in period $t$:

$$\pi(t) = (1 - \alpha)\Psi(\alpha)R(t) - [\bar{c} - \alpha(\bar{c} - c_1)]R(t) = (1 - \alpha)\Psi(\alpha)R(t) - \rho\pi(t - 1). \tag{9}$$

Substitute Eq. (7) into Eq. (8) to obtain $R(t)$ as a function of the degree of openness, $\alpha$, and firm expenditure on R&D in period $t$:

$$R(t) = \frac{\mathrm{RD}(t)}{\bar{c} - \alpha(\bar{c} - c_1)} = \frac{\rho\pi(t - 1)}{\bar{c} - \alpha(\bar{c} - c_1)}. \tag{10}$$

[12] The dynamics of the two models represent the behaviors of myopic and non-myopic firms, as the forward-looking firm captures a multi-period problem in which the firm re-invests its profits in R&D. Following this rationale, one can also interpret the costs and the revenues in the basic version of the model as present values of cash flow.

[13] Empirical evidence supports the assumption that R&D is a fixed proportion of profits. The intensity of R&D in the French software and information sector is estimated to be 14% of their sales and is relatively stable over time (Abi-Saad *et al.*, 2001).

Substitute $R(t)$ into Eq. (9) to obtain an intertemporal link between the firm's profits in successive periods:

$$\pi(t) = \left[ \frac{(1-\alpha)\Psi(\alpha)\rho}{\bar{c} - \alpha(\bar{c} - c_1)} - \rho \right] \pi(t-1). \tag{11}$$

The firm's goal is to maximize the present value of its stream of profits. If $\delta$ is the discount factor, its objective function is

$$\Pi = \sum_{t=0}^{\infty} \delta^t \pi(t) \tag{12}$$

Equations (11) and (12) show that we can use a simple induction argument to find optimal levels of $\alpha$. In each period the firm's problem is identical to that of the previous period. The profits of last period $\pi(t-1)$ serve only as a scaling factor in the optimization problem of period $t$. On the assumption that the system converges to a steady state, a simple induction argument shows that $\alpha^*$ is constant.

Differentiating $\pi(t)$ with respect to $\alpha$:

$$\frac{\partial \pi}{\partial \alpha} = \frac{-\Psi(\alpha)c_1}{(\alpha c_1 + (1-\alpha)\bar{c})^2} + \frac{(1-\alpha)(\partial \Psi / \partial \alpha)}{(\alpha c_1 + (1-\alpha)\bar{c})}. \tag{13}$$

If revenues are fixed with respect to $\alpha(\Psi(\alpha) = \text{constant})$, then from Eq. (13) we see that $(\partial \pi / \partial \alpha) < 0$ and it is optimal for a firm to keep all of its software proprietary $(\alpha^* = 0)$.

PROPOSITION 4 *If $\Psi(\alpha)$ is constant, then $(\alpha^* = 0)$.*

This proposition contains the apparent result that firms do not open their source in the short run, in order to make rapid advances, and then reap benefits in the future by closing their source. This is explained by the fact that firms never generate revenues on modules that were developed in the open paradigm. Any module developed as open source is available to all competitors, and thus if there are potential revenues to that feature, all competitors will offer it, thus eliminating it as a source of product differentiation, and so eliminating revenues. Only features developed as closed source contain potential revenues. Rapid advances today do not result in higher revenues tomorrow in this model. For this sort of investment to be possible, the model would have to include some secondary effects of rapid advance that affect the market share of the firm, such as network externalities and a sufficiently rapid increase in installed base.[14]

If the change in revenue per feature compensates for the change in the total income of the firm from modifying the open/closed mix (i.e. changing $\alpha$), then it is possible that it may be optimal for the firm to open its source. From Eq. (13), we have Proposition 5.

PROPOSITION 5

(a) *If*

$$\Psi(0) \geq \left[ \frac{\partial \Psi(0)}{\partial \alpha} \times \frac{\bar{c}}{c_1} \right] \quad \textit{at } \alpha = 0,$$

*then $\alpha^* = 0$.*

[14] On this issue see, for example, Givon *et al*. (1995) and Laffont *et al*. (1998).

(b) *If*

$$\Psi(0) \geq \left[\frac{\partial \Psi(0)}{\partial \alpha} \times \frac{\bar{c}}{c_1}\right] \quad \textit{at } \alpha = 0,$$

*and* $\Psi(1) \leq 0$ *at* $\alpha = 1$, *then* $0 < \alpha^* < 1$.

Finally, in parallel with the analysis of the myopic firm, setting $\partial\pi/\partial\alpha = 0$ in Eq. (13) and solving for $\Psi(\alpha)$ gives

$$\Psi^*(\alpha) = \left[\bar{c} + c_1 \frac{\alpha}{1-\alpha}\right] \Psi_0. \tag{14}$$

This gives us directly the following proposition.

PROPOSITION 6 *If* $\Psi(\alpha)$ *is continuous and*

$$\Psi(\alpha^*) \gtrless \left[\bar{c} + c_1 \frac{\alpha^*}{1-\alpha^*}\right] \Psi_0 \quad \textit{at } \alpha = \alpha^*,$$

*then a firm operating with* $\alpha = \alpha^*$ *will increase* (*decrease*) *its level of openness.*

Propositions 2 and 5 and Propositions 3 and 6 give different results for optimal amounts of openness and the critical openness–revenue relationship. These differences are driven by the fact that in the myopic case (Propositions 2 and 3) the degree of technical advance is fixed each period and the R&D expenditure varies, whereas in the non-myopic case, since R&D expenditures are determined by last period's profits, the reverse is true.

Notice that if revenues are fixed with respect to openness, technical advance is at a minimum $(\partial\pi/\partial\alpha) < 0$ in Eq. (13); and technical advance of each period, $R(t)$, is determined by profits of the previous period: $R(t) = \rho\pi(t-1)$. However, if the goal is to *maximize technical advance*, the optimal development mode is *entirely open source*, which is approached if revenues increase rapidly with openness. It is almost certainly the case, though, that maximizing technical advance is sub-optimal due to its effects on accelerating depreciation. Generating an interior amount of openness is desirable from a policy maker's perspective (driven by the view that some intermediate rate of technical advance will maximize social welfare) and has to be supported by a complimentary pricing mechanism whereby the effective revenue to a firm from an advance in its software's technical quality increases with openness, but only to a point. This is likely to be a non-trivial policy problem.

## 4 DISCUSSION

We have seen the strong result in the model as described: The negative relationship between the degree of openness and profits implies that a profit maximizing firm will keep all its code proprietary. Yet, application of open-source methodology gains popularity among commercial firms, which publicly release the source code of their products. What is their motivation?

From Proposition 5 we see that if the revenues of the firm increase with openness a profit-maximizing firm will open its source. In the model as developed, the increment to technical quality, $R(t)$, is determined simply by R&D expenditure and the degree of openness since open source and proprietary software have different costs. But if open source has an independent effect on quality, if it is more reliable for example, as some suggest, then this would support an increase in revenue as the degree of openness increases.

A second explanation has to do with quantities. Implicitly, $\Psi(\alpha)$ is the revenue attributed to the current increment in technical quality. If an increase in openness increases sales, then $\Psi(\alpha)$ increases with $\alpha$. This is made stronger by the observation that for software goods, marginal cost is essentially zero, so average costs fall with output. The motivation for a firm to open its source code (apart from potential threats from the Justice Department) can be to increase sales. In a larger model, in which network externalities would play a role in a competition for market share, firms are more motivated to choose open-source strategies, as an increasing market share could have the positive effect of locking a competitor out of the market and providing a monopoly in the future.

### 4.1 An Extension

In the model developed above, it was not possible for a firm to capture revenues from anything developed in the open-source mode. It could be, though, that due to reputation effects or packaging effects, a firm may be able to generate revenues on these features. This can be captured in the model by introducing the parameter $\Psi_1$. It is reasonable to assume that while $\bar{c} > c_1$ similarly $\overline{\Psi} > \Psi_1$. Clearly, if $\overline{\Psi} - \bar{c} < \Psi_1 - c_1$ then the optimal strategy for any firm is to embrace open source for all of its development. The more interesting case is when $\overline{\Psi} - \bar{c} > \Psi_1 - c_1$. In this case the analysis follows in a straightforward way, re-writing the revenue definition in Eq. (6) as $\mathrm{RV}(t) = (\alpha\Psi_1(\alpha) + (1-\alpha)\Psi_2(\alpha))R(t)$. Still, however no interior solution exists when revenues do not change with the degree of openness.

This article has presented a lower bound on the responsiveness of revenues to the degree of openness for software firms that choose strategic application of open source for their products. Critical in this bound is the difference in the costs of developing a module as open versus closed source for the firm. It is important to remember here that open-source development is not free, even though a significant proportion of the inputs are. We obtain a knife-edged result such that if the revenue increases rapidly enough with openness, the optimal strategy changes from entirely proprietary to a strictly positive degree of openness. Like most knife-edge results, this one is extreme, but it does suggest that following certain environment changes in which it becomes possible to capture more revenues from open source or a reduction in the costs of open-source development, we would expect to see firms that have historically been developing only proprietary software moving aggressively towards open source.[15] We have seen this strong move from proprietary software to open source with, among others, IBM's Eclipse, Netscape's Mozilla project, and Sun's move from Star Office to create Open Office. Using this idea as a base for further analysis, it is possible to pursue the investigation of open source as a firm strategy. Understanding why firms engage in this strategy then devolves to understanding time mechanisms by which this strategy increases revenues or how the responsiveness of revenues rises above the threshold.

***Acknowledgements***

The authors are grateful to Maria Brouwer, Aija Leiponen, Gottfried Leibbrandt, the participants of the DRUID Summer Conference (Copenhagen, June 2002), and to three anonymous referees for useful suggestions and comments.

[15] Note that incentives of firms to invest in R&D are not endogenized in this model.

## *References*


Abi-Saad, P., David, C., Gandon, M. and Weisenburger, E. (2001) *Recherche & Developpment en France: Resultats 1999, Estimations 2000, Objectifs Socio-Economiques du BCRD 2001*. Paris: Ministere de l'Education Nationale.

CNET News (2001) IBM makes $40 million open-source offer, November 2001. Available at: http://news.com.com/2100-1001-275388.html (accessed October 2003).

Conner, K.R. and Rumelt, K.P. (1991) Software Piracy: An Analysis in Protection Strategies. *Management Science*, **37**(2), 125–139.

Cowan, R. and Harison, E. (2001) *Intellectual Property Rights in a Knowledge-Bosed Economy*. MERIT Study for the Dutch Advisory Council for Science and Technology Policy (AWT), AWT Background Study No. 21, May 2001, Maastricht.

Givon, M., Mahajan, V. and Muller, E. (1995) Software Piracy: Estimation of Lost Sales and the Impact on Software Diffusion. *Journal of Marketing*, **59**(1), 29–37.

Granstrand, O. (2000) The Shift Towards Intellectual Capitalism – The Role of Infocom Technologies. *Research Policy*, **29**(9), 1061–1080.

Grossman, C.M. and Helpman, E. (1991) *Innovation and Growth in the Global Economy*. Cambridge, MA: MIT Press.

Hertel, G., Niedner, S. and Herrmann, S. (2003) Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel. Forth-coming in Research Policy, Special Issue on Open Source Software Development.

Laffont, J.J., Rey, P. and Tirole, J. (1998) Network Competition: Overview and Nondiscriminatory Pricing. *RAND Journal of Economics*, **29**(1), 1–37.

Lakhani, K. and von Hippel, E. (2000) How Open Source Software Works: "Free" User to User Assistance". MIT Sloan School of Management Working Paper, No. 4117, May 2000.

Lerner, J. and Tirole, J. (2002) Some Simple Economics of Open Source. *Journal of Industrial Economics*, **52**(2), 197–234.

Lessig, L. (2002) Open Source Baselines: Compared to What? In Hahn, R.W. (ed.) *Government Policy toward Open Source Software*. Washington, DC: AEI-Brookimings Joint Center for Regulatory Studies.

McKelvey, M. (2001) The Economic Dynamics of Software: Three Competing Business Models Exemplified Through Microsoft, Netscape and Linux. *Economics of Innovation and New Technologies*, **10**(3), 199–236.

Nordhaus, W.D. (1969) *Invention, Growth and Welfare: A Theoretical Treatment of Techno-logical Change*. Cambridge, MA: MIT Press.

PC Magazine (2001) Performance Tests: File Server Throughput and Response Times. Novem-her 2001, Available in: http://www.pcmag.com/article2/0,4149, 16227,00.asp (accessed October 2003).

PC Magazimie (2002) Samba Runs Rings Around Win2000. April 2002. Available in: http://www.vnunet.com/News/1131114 (accessed October 2003).

Reese, B. and Stenberg, D. (2001) Working Without Copyleft. O'Reilly Network, Available in: http://www.oreillynet.com/pub/a/pohicy/2001/12/12/transition.html (accessed October 2003).

Rothman, J.B. and Buckman, J. (2001) Which OS is Fastest for High-Performance Network Applications? *Sys Admin*, **10**(7), 15.

Shy, O. and Thisse, J.F. (1999) A Strategic Approach to Software Protection. *Journal of Economics and Management Strategy*, **8**(2), 163–190.

Smith, B.L. (2002). The Future of Software: Enabling the Marketplace to Decide. In: Hahn, R.W. (ed.) *Government Policy toward Open Source Software*, Washington, DC: AEI-Brookings Joint Center for Regulatory Studies.

Young, R. (1999) Giving It Away: How Red Hat Software Stumbled Across a New Economic Model and Helped Improve an Industry. In: DiBona *et al*. (eds) (1999) *Open Sources: Voices from the Open Sources Revolution*, Sebastopol: O'Reilly and Associates.