Mechanism Design to Promote Free Market and Open Source Software Innovation

Geoffrey Parker
Tulane University
New Orleans, LA 70118
Email: gparker@tulane.edu

Marshall Van Alstyne
MIT Center for eBusiness
Cambridge, MA 02139
Email: marshall@mit.edu

Abstract— **Some economic strategists now assert that the greatest value in information goods is not created by thestrongest and most restrictive intellectual property protection. Proponents of Open Source Software argue for value created by peer review and openly modifiable shared code. To explore these ideas, we articulate a balance of incentives as indexed by the length of time that software remains proprietary, and openness as indexed by the amount of the platform code base that an author releases to the developer community (and users) to promote the creation of new products. We analyze the trade-off between early and late release based on a two-sided network externality that explores how the release of free information benefits those who develop as well as those who consume. We also introduce a framing innovation that places existing licenses in a space that suggests where unexplored socially optimal licenses might exist.**

I. Introduction

Intellectual property law embodies the principle that incentives enhance social welfare by granting temporary monopolies to authors and inventors as a reward for innovation. Recent developments, however, challenge one prevailing interpretation of the idea that proprietary systems create the greatest value. The challenge appears at one level among economists and legal scholars who assert that the greatest value in information goods is not created by the strongest intellectual property protection (Lemley, 2004; Lessig, 2001; Shapiro & Varian, 1999). It appears in another form among proponents of free software, who argue for greater social cohesion and for access freedom as a right (Stallman, 1992), and among proponents of the open software movement, who argue for value created by peer review, reuse, and complementary investment (Raymond, 2000).

This research articulates a balance between proprietary incentives and access freedoms that can promote both innovation and widespread distribution and network externality benefits from open access. The proposed mechanism is quasi-open—adjustable between fully open and fully closed. It behaves as fully open in the sense that part of a platform is (1) freely available, (2) open to inspection, (3) modifiable, and (4) redistributable. This generates network externality and access benefits. It behaves as proprietary in the sense that any modified or derivative works may be sold but are subject to disclosure requirements. This restores profits, price signals, and incentives. Like free / open source software, derivative works are subject to the four freedoms noted above, but unlike standard free software, these requirements do not bind for a brief proprietary period, specified by the platform author, during which any 3^{rd} party developer may exercise pricing power based on the value of an innovation.

In practical terms, this framework generalizes a range of alternative technologies: fully closed systems such as those governed by restrictive "end user license agreements" (EULAs), partially open systems such as those exposing "applications programming interfaces" (APIs), and even fully open systems such as games where platform adoption is subsidized and profits derive primarily from royalties on the sale of third-party enhancements.

The intuition for this model proceeds specifically from software—and thus operates primarily under copyright—but it applies potentially to any innovation with two properties: (i) the intellectual property represents a platform on which subsequent innovation depends and (ii) property rights and contract law can alter the terms of access and subsequent disclosure requirements placed on derivative works. Both properties exist for operating systems, gaming platforms, audio and visual platforms, and any form of software that allows 'plug-ins.' To some extent, however, these properties also exist for hardware platforms, credit card networks, telecommunications infrastructure, and biotechnology.

Following a review of related literature, this paper develops an analytic model of tradeoffs among the welfare interests of the original platform author, the users, and third party developers. We show how too much protection can reduce welfare and hurt the interests of even profit-motivated participants. We also analyze the welfare effects of competing kinds of contracts, including those designed to maximize freedom. Achieving the best of both free market and free software then requires careful mechanism design. We conclude with a discussion of broader implications.

II. Literature

Related work on the economics of intellectual property examines the question of optimal patent length and breadth, finding that property rights should be long-lived (infinite) and narrow when the flow rate of monopoly profits causes welfare losses (Gilbert & Shapiro, 1990; Klemperer, 1990). Alternatively, they should be short-lived and broad when consuming inferior substitutes is the principal cause of welfare

losses (Klemperer, 1990). In contrast, our work considers the original innovator's interest in subsequent developments, a factor that leads to finite-lived protection. Our mechanism places the original author in a position analagous to that of a social planner. And while a social planner chooses a shorter proprietary period than the platform author, the author substantially internalizes the social gains from innovation and therefore chooses a horizon on first round innovation that brings forward the profits from second round innovation.

This work complements the literature on sequential innovation, which argues for longer property rights when innovation proceeds in rounds led by different firms (Green & Scotchmer, 1995). If property rights are short, firms appropriate little of the social value of derivative innovations and underinvest in basic research. Our work takes the next practical step and asks, given a platform and potential for sequential innovation, what mechanism offered to follow-on developers maximizes profits and welfare. If network effects are small or the ability to build on the platform is small, we find that the author prefers to go it alone. If, however, either the coefficient of reuse or network effects are large, then the author prefers to engage complementary investment via opening the platform even at the cost of losing profits on the platform itself.

Openness of the platform, coupled with openness of derivative works upon expiration of the proprietary period, offers benefits analogous to those of patent pools. With cumulative innovation, strengthening property rights in mutually blocking technologies can have the perverse effect of stifling innovation (Shapiro, 2001). Firms that cross-license and make the collection of property rights available as a pool avoid this and other difficulties (Shapiro, 2001; Rey & Tirole, 2003). Relative to a patent pool, the open platform and subsequently open derivative works also allow users and developers to treat the asset as common resource. It reduces problems of multiple marginalization; and it limits transaction costs of negotiating in the context of multi-party hold-up. Openness with default access also avoids problems of exclusive dealing and cartels that are downside risks of patent pools.

A commitment to open the platform, and keep it open into perpetuity, parallels insight from the second-sourcing literature (Farrell & Gallini, 1988). If users incur high asset-specific setup costs, and buy goods either in multiple periods or as components of a system, then a monopoly provider of a platform technology benefits from forward competition. User willingness to adopt ex ante is conditioned by commitments to forgo exploitation ex post. Thus a commitment to an open architecture, or to license to competitors, provides a means of growing the base of users who would otherwise rationally limit their consumption, prefering to avoid lock-in and future price gouging.

The innovation and legal literature (Hippel & Krogh, 2003; Benkler, 2002; Lemley, 2004) consider competing models of "private investment" and "collective action" for producing new works. The former focus on private property and efficient intellectual property protection in the context of free markets. The latter focus on collaboration among innovators to produce public goods in the context of such market failures as positive externalities, non-excludibility, and the absence of price signals. We seek to model the tradeoffs inherent in this debate in an effort to clarify and strengthen claims regarding the beneficial uses of intellectual property. We show, for example, that if developers are motivated by access freedom, then more liberal licensing yields greater welfare in not only relative but also absolute terms. That is, welfare from "collective action" matched with a free access license exceeds that of "private investment" matched with a proprietary license.

Managing these tradeoffs effectively requires a platform author to consider how much current access to allow while also setting terms for downstream use. At time zero, authors can choose to release some fraction of their platform in order to foster immediate adoption and encourage downstream innovation. They can also set a future date at which derivative intellectual property becomes available. In a tangible context, this resembles the 18^{th} century decision by railroads to allow developers to build on railroad land but then give up these improvements after a suitable time had elapsed. Granting free access immediately, however, contributes nothing to a platform author's profits. Access could have been sold. Tension arises in giving up access to encourage development. Moreover, this same tension arises in promoting subsequent innovation among 3^{rd} party developers. Promoting adoption by earlier release discourages innovation more than later release.

III. Theoretical Foundations & Model

We analyze the trade-off between early and late release based on two novel approaches. The first is a network externality twist that explores how the release of free information benefits those who develop as well as those who consume. The second is a framing innovation that places existing licenses in a space suggesting where unexplored socially optimal licenses might exist.

The network externality model extends Katz and Shapiro (1985) to more diverse and complementary markets. Although a recent area of study, two-sided network effects have begun to receive considerable attention (Parker & Van Alstyne, 2000; Caillaud & Jullien, 2001; Rochet & Tirole, 2003; Armstrong, 2002; Parker & Van Alstyne, 2002). These represent a demand economy of scale that crosses markets as distinct from one that stays within the same market. The presence (or absence) of each side makes the other more (or less) valuable to an organization that deals with both halves at once. Here, the author of an original software platform can look to consumers and other developers as these two halves.

In the present context, author A develops a platform on top of which software developers D can create enhancements that add value for end-users or consumers C. Author A then chooses an amount σ of platform code to open in order to stimulate developer participation. The author also chooses a time t after which developer enhancements must be folded back into the open code base. The more that developers are motivated to create enhancements, the greater the benefits to C. Similarly, the larger the C market, the more attractive it

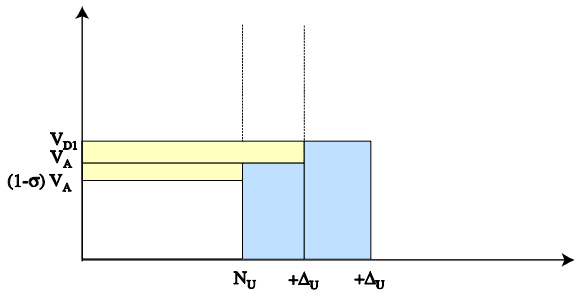

Fig. 1. A fraction, σ, of the original platform value, V_a is made freely available, leaving direct platform profit of $(1-\sigma)\,V_a$. However, the release of code allows developer enhancements, which creates value and brings additional consumers to the platform.

is for developers to code for that market, contingent on their access to source code. This is consistent with the size of a problem scope influencing the prestige associated with solving it (Raymond, 2000). The computer games market represents another example where the presence of a large number of free game levels makes it attractive to consume particular games; and the presence of a larger gaming user community makes it more attractive to develop for a particular game. Similarly, for operating systems, the larger the installed base of users, the more attractive is developing applications for that OS; and the more applications, the more attractive it is to consume that OS.

A. Timing

To capture the tension between early and late release, we begin with a simple two period model that includes the value of the platform before developer participation, the first period developer enhancements, and the second round of developer enhancement that builds on the first round after the release period t. Timing is as follows. Period 0 - Platform author chooses (i) fraction (σ) of an original innovation to freely release and (ii) the incentive period t after which downstream developer enhancements are completely and freely released to the code base. Period 1 - Developers add value by creating derivative works that build on the original open code base (value determined by $\sigma \times$ original value created by the platform author) and incentives captured in t. Period 1 - Consumers observe developer value creation and endogenously choose consumption levels. Period 2 - In a fulfilled expectations equilibrium, developers observe consumer participation in period 1 and choose participation and value-adding levels having gained access to the open code freely released at the end of period 1.

Consumers know that developer enhancements will be freely released under the open regime after period t. In a Coasian consumption problem, this conditions their willingness-to-pay in the current period on their discount rate r. We assume that consumers share a common value for the product. Although made for tractability, this assumption enjoys theoretical support from Bakos and Brynjolfsson (1999) who show that a point estimate of consumer value is reasonable for relatively large bundles of information goods. If a developer adds value v, this leads simply to choosing price such that $p \leq v(1-e^{-r\,t})$. Note that the longer the proprietary period, the more the consumer is willing to pay for a particular developer enhancement. When the proprietary period is very short, the consumer will pay very little since she knows that the product will soon become freely available. In the analysis below, we let $\delta = e^{-r\,t}$.

B. Developer Problem

We assume that developers produce in direct proportion to the freely open code base and in response to the length of time they can benefit from their effort. This allows incentive effects to influence code development (below, we consider developers who are motivated by the degree σ to which the original platform is freely open). In any period, developer output y increases in both t and in the open code base Ω. Let Ω_t represent open code at time t, let k be a re-use coefficient that scales the ability to use open code in new production, and let $L(\Omega_t, t)$ be labor or effort as a function of costs and the time to recover investments. Then code created by a developer is $k\,\Omega_t\,L(\Omega_t, t)$. We capture the incentive effect of time by replacing $L(\Omega_t, t)$ with a concave increasing function of time, $(1-\delta)$. The total code created by all of the (symmetric) developers in period t is then $q_{d\,t}\,k\,\Omega_t\,(1-\delta)$. Note that we assume that consumption of one developer's enhancement does not affect the value realized by consuming another developer's enhancement. Hence we assume that the value created by a developer is the same as the amount of code created. That is, $v_t = k\,\Omega_t\,(1-\delta)$. Total value created by all developers is then the same as total code created, $q_{d\,t}\,k\,\Omega_t\,(1-\delta)$. This is consistent with our treatment of code (and value) created by the platform author.

C. Platform Author Problem

The platform author chooses two parameters. The first is the sharing parameter, σ, that governs the fraction of the original platform code to be freely released. This fraction is no longer purchased but is freely adopted by consumers. The second choice parameter is the proprietary time period t that generates revenues for 3^{rd} party developers on any enhancements they create. Before t expires, enhancements are profitable but consumer adoption is also limited via positive prices. Upon expiration of t, developers contribute derivative works back to the open code base from which they drew their original sources. We explore optimal choices for t and σ based on different welfare maximizing criteria, including combinations of freedom of access, adoption rates, and revenues. Since consumers face a Coasian consumption tradeoff, the original firm must factor consumers' strategic behavior into determining the optimal time to release the code under open code terms. Forcing early release of enhancements creates consumer surplus at the cost of reducing developer incentives.

Our point of departure is to make explicit the chicken and egg relationship between customers and developers. To

model this relationship, we use a two-sided network externality framework. The network externality term e_{du} measures how much effect the presence of developers has on the size of the consumer market. Conversely, e_{ud} determines how much effect the presence of consumers has on the size of the developer market.

D. User and Developer Participation

The logic to determine the number of users and developers is that the more users who join the platform, the more developers want to produce content for the platform, and hence the more users who want to join. This feedback generates an infinite sequence that defines the externality between users and developers. Let N_u be the core market for the base platform before any developer enhancements are added. Let N_d be the core number of developers for the base platform who come to the platform independent of the number of users.

The first term of the infinite sequence is the increase in the number of developers, $\lambda\,\Omega$ where λ is the conversion rate of open code to new developers for the platform. The second term is the increase in the number of users as function of the increase in the number of developers, $e_{du}\,\lambda\,\Omega$ where e_{du} is the externality from developers to users. The third term is the increase in the number of developers as a function of incremental users, $e_{ud}\,e_{du}\,\lambda\,\Omega$, where e_{ud} is the externality from users to developers. The sequence then continues with the interpretation that the odd terms are the increase in developers, and the even terms are the increase in users.

The total increase in the number of developers can then be found by summing up the odd terms: $\lambda\,\Omega\,(1 + e_{ud}\,e_{du} + (e_{ud}\,e_{du})^2 + (e_{ud}\,e_{du})^3 + ...)$. So long as $0 \leq e_{ud}\,e_{du} < 1$, this converges to $\frac{\lambda\,\Omega}{1-e_{ud}\,e_{du}}$. The total increase in the number of users can be found by summing up the even terms: $e_{du}\,\lambda\,\Omega\,(1 + e_{ud}\,e_{du} + (e_{ud}\,e_{du})^2 + (e_{ud}\,e_{du})^3 + ...)$. Again, with the restriction that $0 \leq e_{ud}\,e_{du} < 1$, the even terms converge to $\frac{e_{du}\,\lambda\,\Omega}{1-e_{ud}\,e_{du}}$). Letting $M_u = \frac{e_{du}\,\lambda}{1-e_{ud}\,e_{du}}$ and $M_d = \frac{\lambda}{1-e_{ud}\,e_{du}}$, we define the total number of users and developers as

$$q_u = N_u + M_u\,\Omega \tag{1}$$

$$q_d = N_d + M_d\,\Omega. \tag{2}$$

In the two period model, quantities are subscripted $q_{u1}\,, q_{u2}\,, q_{d1}\,, q_{d2}$ and the open code base is subscripted Ω_1 and Ω_2. The term N_d can be interpreted as the number of developers who would participate in the platform independent of any motivation to profit from user participation.

E. Free Code Base

In the two period model we analyze, the free code base evolves as follows. In the first period, the platform author can choose what proportion, σ of the original platform to make immediately free. This free code base provides the foundation upon which developers are able to create new additions to the platform. In the second period, in addition to the code released in period one, the new code that was created by the developer community in the first period is freely and completely released after the proprietary period, t. The developers' ability to reuse code is captured by the parameter k. The total amount of code created increases in the total number of developers who join the platform in the first period. As noted above, developer output increases in the length of time over which developers can charge users for enhancements.

$$\Omega_1 = \sigma\,\pi_a \tag{3}$$

$$\Omega_2 = \Omega_1 + k\,q_{d1}\,\Omega_1\,(1-\delta) \tag{4}$$

In order to facilitate the analysis of welfare and profit as a function of time, we introduce a useful transformation between variables and their first derivatives with respect to time.

Lemma 1: The time derivative of second period quantity (for both users and developers) is the difference between first and second period quantity times $\frac{r\,\delta}{1-\delta}$. That is, $q_2' = \frac{r\,\delta}{1-\delta}(q_2 - q_1)$. Because $q_2 - q_1 \geq 0$, $q_2' \geq 0$. Identical to quantity, the time derivative of the second period open code base is also the difference between first and second period code base times $\frac{r\,\delta}{1-\delta}$. That is, $\Omega_2' = \frac{r\,\delta}{1-\delta}(\Omega_2 - \Omega_1)$. $\Omega_2 - \Omega_1 \geq 0$ implies $\Omega_2' \geq 0$.

Proof: Follows directly from definitions of q_{d2}, q_{u2} and Ω_2. ■

F. Welfare Analysis

Total welfare in the two periods is made up of author profit, developer profit, and consumer surplus. The second period is discounted by factor δ back to the first period. Detailed expressions for the components of each are given below.

Author Profit

$$q_{u1}\,(1-\sigma)\,\pi_a \;=\; \text{Per 1 platform sales} \tag{5}$$

$$\delta\,(q_{u2} - q_{u1})\,(1-\sigma)\,\pi_a \;=\; \text{Per 2 platform sales} \tag{6}$$

$$\phi\,p_1\,q_{d1}\,q_{u1} \;=\; \text{Per 1 license fees} \tag{7}$$

$$\delta\,\phi\,p_2\,q_{d2}\,q_{u2} \;=\; \text{Per 2 license fees} \tag{8}$$

The first term of author profit multiplies the number of users times the amount of value given away by the platform author. The second term, discounted back to period one, multiplies incremental users times fraction of value given away by the platform author. The third and fourth terms are the fraction, ϕ, of developer sales to consumers that is paid by the developer to the platform author as a license fee.

Developer Profit

$$(1-\phi)\,p_1\,q_{d1}\,q_{u1} \;=\; \text{Per 1 profit} \tag{9}$$

$$\delta\,(1-\phi)\,p_2\,q_{d2}\,q_{u2} \;=\; \text{Per 2 profit} \tag{10}$$

Total developer profit is made up of sales of developer enhancements to consumers less the license fee paid to the platform author.

Consumer Surplus (CS)

$$q_{u1}\,\sigma\,\pi_a \;=\; \text{Per 1 platform}$$

$$\delta\,((q_{u2} - q_{u1})\,\Omega_2 + q_{u1}\,(\Omega_2 - \Omega_1)) \;=\; \text{Per 2 platform}$$

$$(v_1 - p_1)\,q_{d1}\,q_{u1} \;=\; \text{Per 1 developer}$$

$$\delta(v_2 - p_2)\,q_{d2}\,q_{u2} \;=\; \text{Per 2 developer}$$

The first term of consumer surplus measures the surplus from consumption of the portion of the platform that is given away. The second term measures consumption of second period open code Ω_2 by incremental users, $q_{u2} - q_{u1}$, plus consumption of incremental open code, $\Omega_2 - \Omega_1$, by first period consumers.

1) Total Welfare: The expression for total welfare is simpler than the elements above might suggest because wealth transfers have no effect and all σ and $(1-\sigma)$ terms, and ϕ and $(1-\phi)$ terms collapse together. Total welfare can be expressed as

$$q_{d1}\, q_{u1}\, v_1 + \delta\, q_{u2}(q_{d2}\, v_2 + \Omega_2) + \pi_a\,(q_{u1} + \delta\,(q_{u2}\,(1-\sigma) - q_{u1})). \quad (11)$$

The first term is period one developer additions. The second term is second period developer additions plus the value of open code. The third term is first period consumer surplus from the platform (which combines author profit and consumer surplus) plus second period new user surplus less period one double-counting.

In general, the social planner prefers to make the original platform freely available, but prefers an intermediate value for time to release of derivative works. An example of this result can be seen graphically in Figure 2. We show the general case below in Proposition 2.

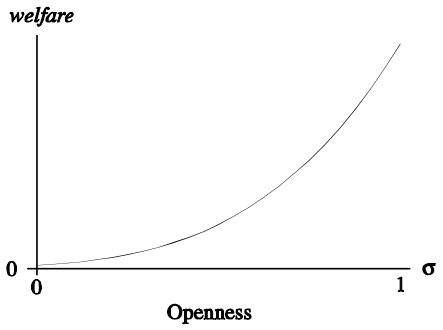

Fig. 2. Welfare as a function of openness.

Figure 3 shows that the social planner prefers an intermediate value for time to release of derivative works.

Proposition 2 (Social planner preferences over σ and time): To maximize welfare, the social planner's optimal contract $[\sigma^*, t^*]$ is a corner solution in σ but an interior (finite) value in t. This result holds for any level of network externality, including zero.

Proof: Substitute $v_1 = k\,\Omega_1(1-\delta)$, $v_2 = k\,\Omega_2(1-\delta)$, $\Omega_1 = \sigma\,\pi_a$, and $\Omega_2 = \sigma\,\pi_a + q_{d1}\,k\,\pi_a\,(1-\delta)$ into equation 11. After grouping terms, note that each element has a non-negative derivative with respect to σ, thus establishing the first part of the proposition. To establish the zero network externality result, substitute $M_u = 0, M_d = 0$ into 11, take the derivative with respect to σ and note that the result is non-negative.

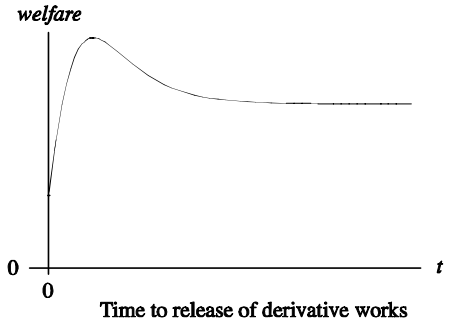

Fig. 3. Welfare as a function of time to release of derivative works.

To establish the claim with respect to time, take the derivative of total welfare with respect to t and apply Lemma 1. Note that the derivative of welfare with respect to t at $t = 0$ is positive. At the other end of the time scale, take the derivative of welfare with respect to δ and evaluate $\delta = 0$ (where $t \to \infty$). The derivative is positive (for any level of network externality) which implies a benefit to lowering t. ■

G. Platform Author Choices

We now explore the platform author's preferences over t and σ. The platform author is concerned with optimizing its subset of welfare components over both periods. The platform author's preferences are less general than a social planners and depend upon specific parameter values. We summarize these preferences in the following proposition. Note that limiting values are evaluated instead of closed form solutions. Although it is possible to obtain closed form solutions using mechanical means, the resulting expressions are too lengthy for meaningful interpretation.

Proposition 3 (Platform author preferences over σ and time): The platform author prefers greater openness for large values of code reuse, k, network externality, M_u, and developer population N_d. Similarly, the platform author prefers a finite time to free derivative release for large values of code reuse, k and network externalities, M_u and M_d. When there is no code reuse, platform authors prefer perpetual copyright.

Proof: To establish the result for σ, take the derivative with respect to σ of the sum of the platform author profit elements and evaluate at $\sigma = 0$. The resulting expression is positive (implying a benefit to a greater fraction of freely released platform code) when the following expression holds: $-N_u + k\,\phi\,N_d\,(1 + \delta + k\,N_d\,\delta - k\,N_d\,\delta^2)\,N_u\,(1-\delta)^2 + M_u\,(k\,(1-\delta)\,\delta\,N_d + 1)\,\pi_a > 0$. To establish author preferences over time, take the derivative with respect to δ of the sum of the platform author profit elements and evaluate at $\delta = 0$. The resulting expression is positive (implying an interior t) when $2\,k\,q_{d1} + k^2\,{q_{d1}}^2 + k\,M_d\,(1 + k\,q_{d1})^2\,\sigma\,\pi_a > 1$. Note that when $k = 0$, this expression never holds. Further note that when $M_u = 0, M_d = 0$, the condition for a positive derivative is $k\,N_d > 1$. ■

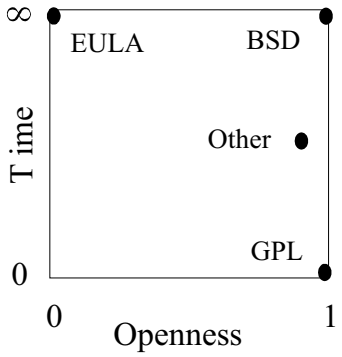

Fig. 4. Contracts mapped by time to free release of derivative works ($0 =$ immediate, $\infty =$ end of copyright) and the fraction of the original platform code base that is freely released ($0 =$ none, $1 =$ all). Note that a 95-year corporate copyright is well approximated by a perpetual (infinite) copyright at a reasonable discount rate.

Note that $-N_u$ is the only negative term in the analysis of preferences over σ. Where there is a large user base that does not depend upon developer enhancements (N_u), the platform author is less interested in releasing free code and instead prefers to capture rents directly from these users. The middle term grows in the royalties the author collects on new code creation. The last term grows in user network effects. Once the new code royalties and network effects become high enough, the platform author benefits from setting a positive sigma.

H. Alternative Licenses

A major contribution of this framework is that it generalizes several different types of contracts. We can then ask which contract optimizes a given welfare criterion under a specific set of parameter values. If we allow different degrees of openness $\sigma \in [0, 1]$, we can model a range of access to source code that spans, for example, closed proprietary systems (EULA), access to application program interfaces (APIs), trial-ware, share-ware, royalty free binaries, open source (BSD), and viral open source (GNU). Similarly, different times to release $t \in [0, \infty]$ can represent disclosure now, after a brief delay, after a long delay, or on expiration of copyright. Several existing licenses are then captured in Figure 4.

Fully open source software, such as that released under the GNU Public License (GPL) releases everything, $\sigma = 1$, and requires subsequent code to be released immediately, $\delta = 0$. In contrast, the Berkeley Software Distribution (BSD) license releases everything but then places no restrictions on subsequent disclosure, $\delta = 1$. The Microsoft End User License Agreement (EULA) provides no open software, $\sigma = 0$, and therefore places no disclosure restriction on derivative works, $\delta = 1$. Interestingly, in this framework, many standard licenses appear as corner solutions. Thus we have sought to define and explore licenses with interior solutions for which $0 < \sigma < 1$ and $0 < \delta < 1$. Such a license might be termed a flexible copyright, or meta license, that incorporates features of both proprietary copyright and open source copyleft.

Proposition 4: Open BSD and GPL licenses never create less total welfare than EULA but unless the royalty rate times code reuse is high enough such that $k\,\phi \geq \frac{N_u}{(N_d + M_d\,\pi_a)\,(N_u + M_u\,\pi_a)}$, a platform author prefers EULA. The relative profits and welfare for EULA, BSD, and GPL style contracts are respectively:

Contract	Author Profit
EULA	$N_u \pi_a$
BSD	$k\,\phi\,\pi_a\ (N_d + M_d\,\pi_a)\ (N_u + M_u\,\pi_a)$
GPL	0
Contract	**Total Welfare**
EULA	$N_u \pi_a$
BSD	$\pi_a\ (N_u + M_u\,\pi_a)\,(1 + k\ (N_d + M_d\,\pi_a))$
GPL	$\pi_a\ (N_u + M_u\,\pi_a)$

Proof: These are found by straightforward substitution of corner solutions EULA: $[\sigma = 0, t = \infty]$, BSD: $[\sigma = 1, t = \infty]$, and GPL: $[\sigma = 1, t = 0]$ into the expressions for author profit and welfare. ■

The open BSD and GPL style contracts always create greater total welfare than the closed EULA license but are not necessarily incentive compatible for the platform author. It is possible that sufficiently large percentage royalties, network effects, or developer value-added make a BSD-style contract a dominant author choice although the actual BSD license does not require or even expect royalties; so this framework presents a more generous case from the author's perspective. Note that if the BSD contract forbids royalties, then the platform author strictly prefers EULA.

1) Freedom motivated developers: It is interesting to note that a BSD contract provides uniformly greater social surplus than the more socially conscious GPL. This result, however, relies critically on the modeling assumption that developers are profit motivated. To probe this further, we examine a model in which developers are motivated by a larger fraction σ of platform code that is released or "freedom". If this is the case, then GPL achieves the highest social surplus of all three licenses. In particular, let α be the fraction of profit motivated developers such that $(1 - \alpha)$ gives the fraction of developers motivated by free access. Then, rewrite the value of developer output $v_t = k\,\Omega_t\,q_{dt}\,(1-\delta)$ to weight effort by relative interest in pricing power and free access. This gives:

$$v_t = k\,\Omega_t\,q_{dt}\,(\alpha(1-\delta) + (1-\alpha)\,\sigma)$$

Under a mix of motivations, the BSD to GPL welfare ratio shifts from $1 + k\,N_d + k\,M_d\,\pi_a$ to $\frac{N_u + M_u\,\pi_a}{N_u + M_u\,\pi_a\,(1 + k\,N_d + k\,M_d\,\pi_a)}$ as the proportion free access motivated developers rises from zero to the full population. This second term is clearly less than 1, showing that a GPL-style contract creates greater welfare in a freedom-motivated context. Assuming that a social planner must offer a single license and that developers exhibit a mix of motives, it is also straightforward to solve for that fraction of freedom motivated developers such that a social planner prefers to switch from BSD to GPL. Interestingly, if developers

are motivated strictly by free access, then a GPL contract is socially optimal not just in relative terms but also absolute terms. That is, welfare exceeds that of a BSD license even when the full population is profit motivated. Setting aside difficult questions on the costs of subsidy and the problem of moral hazard, this advantage parallels certain arguments from the Free Software Foundation made without the benefit of formal analysis.

Welfare parity between these contracts is restored if freedom motivated developers do not exercise the option to exclude others during the proprietary period, $W^{BSD} = W^{GPL}$ if developers are freedom motivated and the proprietary period does no harm. But then a population composed of mixed motive developers benefits from the offer of a proprietary incentive period $W^{BSD} \geq W^{GPL}$. Profit motivated developers have both access and an incentive. Freedom motivated developers have access and the option on profits, which they do not exercise. This leads to a final observation relative to Proposition 4.

Corollary 5: For multiperiod innovation and fraction $\alpha > 0$ of profit motivated developers, author profit and social welfare under a flexible contract are never less than those under BSD or GPL. That is $\pi_a^{CF} \geq \pi_a^{BSD}$, $\pi_a^{CF} \geq \pi_a^{GPL}$ and $W^{CF} \geq W^{BSD}$, $W^{CF} \geq W^{GPL}$.

Proof: Propositions 2 and 3 show that $0 \leq t^* \leq \infty$ is optimal relative to $t^* = 0$ and $t^* = \infty$. ■

I. Private Subcontracting

In addition to the collection of standard open source licenses with $\sigma > 0$, the platform author has the option of subcontracting with specific developers. This can increase profits over those possible from not licensing. There are two primary advantages of subcontracting development relative to open licensing. First, by not opening the code to users, the author need not sacrifice profits on the platform itself. Second, developer contracts might still specify that each must share code with other developers after exploiting the value of their own enhancement. This captures the benefits of a growing code base while allowing developers to charge for the full value of their individual enhancements.

For comparison, let the value of developer output be verifiable and split based on Nash bargaining. This avoids moral hazard and establishes a more compelling case for subcontracting relative to open contracting. Then, to further increase the comparative advantage of subcontracting, allow second generation output to be available at time 0. This simplifies the analysis without affecting incentives, which are no longer based on time. Developers provide two rounds of increased value based on a mutually shared code base that is open with respect to developers but closed with respect to users. Parameters are $t = 0$, $\phi = \frac{1}{2}$, and $\sigma = 1$ but note that there is no profit penalty to the platform author since sharing occurs only privately among subcontractors.

The primary disadvantage of a closed system, however, is that network effects are shut down as closed contracts prevent user access to the underlying code, they disallow modification, and they forbid code sharing. This reduction in network effects can be captured by letting λ, the conversion rate of open code to new developers for the platform, go to zero. Given these parameters, there still exist regions where a platform author prefers to open the code.

Proposition 6: If network effects are absent such that $\lambda \rightarrow 0$, then subcontracting is always a dominant strategy. In contrast, if network effects are present ($Mu > 0$ and $M_d > 0$), then for all σ and δ in the interval $(0, 1)$ there exist postive constants $\bar{M}_d$, $\bar{M}_u$, $\bar{N}_d$ and $\bar{\pi}_a$ such that if any of $M_d > \bar{M}_d$, $M_u > \bar{M}_u$, $N_d > \bar{N}_d$ or $\pi_a > \bar{\pi}_a$ are true, then the platform author prefers an open license.

Proof: Consider first the value of direct platform profits apart from royalties. Substituting parameter values as given for open licensing and closed subcontracting then simplifying the author profit ratio yields $\frac{2(1-\sigma)\,(1+M_u \sigma\, \pi_a\, 1+k\, N_d\;\, \delta(1-\delta)+k\, M_d\, N_u\;\, \delta\,(1-\delta)\,\sigma\,\pi_a))}{2+k\, N_d\,(2+k\, N_d)}$. To prove the first claim, note that either $\lambda = 0$ implies $M_u = 0$, with a ratio of $\frac{2(1-\sigma)}{2+k\, N_d\,(2+k\, N_d)}$. Thus no pair of values $[\sigma, \delta]$ in the valid interval will suffice to increase the ratio above 1. To prove the second claim, let σ and δ be any constants in the range $(0, 1)$ exclusive (such that t lies in the valid interval $0 < t < \infty$). Then all terms in the numerator have positive coefficients. Since M_d, M_u, N_d and π_a exist only in the numerator, increasing any one value arbitrarily increases the ratio above 1. Analagous results hold for royalties alone but note that adding royalties to the open license platform profits would increase its attractiveness relative to a set of closed subcontracts. Ceteris paribus, the choice of optimal values for σ and t increases this effect relative to arbitrary values. ■

That rising network effects and user populations should favor open rather than closed licensing is perhaps not surprising. What is more novel is that given any positive level of network effects, a more valuable platform may itself justify giving a fraction away in order to increase author profits.

IV. Discussion

Having formally modeled several propositions on social welfare and optimal contracts, we turn to a broader discussion of implications and the design of practical mechanisms. The key argument is that there exist economic reasons why a profit maximizing firm would move from a strict proprietary license in the direction of a free or open source license. Conversely, there exist reasons why a welfare maximizing social planner would move from a strictly free license in the direction of a proprietary license. Accordingly, the following discussion articulates a flexible or 'meta' contract that exhibits properties of both closed and open licenses but at different times.

A. Open source destruction of 3^{rd} party developer incentives

Issue: While standard open source licenses do not require developers to license their enhancements at cost, that is their economic effect. If developers must give users both the enhanced code and the right to redistribute, then developers must compete with perfect zero-marginal cost copies of their

own goods. Such markets cannot sustain positive prices above transactions costs on the good itself. Although developers have attempted business models based on indirect compensation, such as services, they have not sustained prices that reflect the economic value of their innovations. For economies not based on gift exchange, this leads to reduced social welfare characterized by under-investment in innovation.

Resolution: One mechanism is to give developers pricing power in their enhancements by delaying the time until the right to redistribute a copy vests with users. This leads to declining pricing power over time as the open period approaches. The length of delay in rights to redistribute, i.e. a proprietary period of an enhancement, should set the area of the demand curve under developer price proportional to the size of investment one wants to call forth while also accounting for the opportunity cost of the developer's time. The key idea is that by allowing 3rd party developers to maintain certain rights in their own enhancements, the original author can attract deeper and more sustained complementary investments.

Note that the offer of a proprietary period places no obligation on the developer to use it. Rather it presents an option to be used at the developer's discretion and he or she may choose the traditional open source gifting role, contributing back to the common code base at time zero or at any time that simply recovers costs. It does, however, create an economic incentive for those who might wish to profit from their own innovation.

B. Information asymmetry on behalf of licensees & the monopsony problem

Issue: For many proprietary licensing opportunities, the developer of an idea may need to disclose his or her concept to the original author in order to obtain both access to the source code and permission to create a derivative work. As owner, however, the author is under no obligation to license and, having learned of the opportunity, may simply refuse to license and then exploit the opportunity without the developer. This is the monopsony problem in which the existence of only one possible buyer creates severe problems of hold-up and inefficiently modest levels of trade.

Resolution: A virtue of the open source software licensing is the offer of a default contract requiring no review by the original author. Any person with an idea can anonymously secure both access to source code and permission to enhance it without disclosing private knowledge. The solution is thus to offer a public default contract, available to anyone, provided that developers meet other reasonable terms of the license.

Note that another version of this solution exists already in the form of applications program interfaces (APIs). Much proprietary code exposes APIs to programmers in order that developers can execute function calls on other software. The current proposal is to extend this proprietary model in the direction of open source such that subroutines can be modified and reused and not merely called.

C. Code forking & incompatibility

Issue: Certain licenses, notably BSD, create incentives to introduce code versions that are incompatible. This arises particularly in cases where developers are allowed to keep their code separate into perpetuity. Forked code bases reduce consumer surplus via a reduction in standardization and network effects.

Resolution: Require developers to license their enhancements to the author in order that improvements are folded back into the common code base. Note that developers can continue to sell if they wish but it becomes economically unattractive. The author will then use economic principles of bundling to discourage incompatible versions. These create barriers to competing product entry but they are not insurmountable.

Of interest is the fact that it remains contractually feasible to fork the code base. In order for this a forking strategy to succeed, however, the economic value of the new offering must be sufficiently great that a developer must add sufficient functionality and offer enough competing bundles that the forked alternative can survive in the market. This permits a radical innovation to evolve while forcing it to pass a threshold test of value that has not simply been contractually excluded.

D. Competition & hold-up in the value chain

Issue: Technology markets frequently exhibit either 'component competition' or 'systems competition.' In the former, an innovator can compete on the basis of specializing in a uniquely low cost or high value part where they have an advantage. In the latter, an integrator competes by offering a high value collection without necessarily offering the best-of-breed or lowest cost for any part. The economic consequence is that specialized providers suffer hold-up by other bottleneck suppliers in the value chain, while integrators suffer technological obsolescence and strategic defection by specialist suppliers from whom they purchase. The crux is that individual firms cannot perpetually provide the best of every part but fragmentary rights distributed among multiple innovators create welfare losses through multiparty bargaining.

Resolution: Software permits near zero marginal cost transfer of the enabling technology throughout a value chain via access to source code. A successful mechanism might therefore offer default rights to the community of developers after the developer of an enhancement has been compensated both for the costs of innovating and the opportunity cost of effort. This simply occurs through the termination of the proprietary period and the commencement of the open source period. The most successful or 'best-of-breed' enhancements become part of the common code base while reducing economic distortions caused by multiparty bargaining.

E. Free riding by 3^{rd} party developers

Issue: The creation of a public good, one that is nonrival and nonexcludable, simultaneously introduces incentive compatibility problems due to free riding. In this case, a third

party developer might wish to invest in an enhancement but also to dishonor the principle of releasing source code upon expiration of the proprietary period.

Resolution: The author's task is to offer developers sufficient value through the open code base that a developer chooses to create a derivative work in preference to incurring the cost of a 'clean room.' This conditions the offer of developer profits collected during the proprietary period to be not less than those net of (i) the higher cost of a clean room and (ii) the lower cost of reusing common code.

Note that another common problem of public goods, overgrazing or the 'tragedy of the commons' does not arise in the case of software due to zero marginal cost reproduction.

F. Strategic misuse of proprietary code by the author

Issue: If an author maintains a key complement as inaccessible proprietary code, then the author can potentially appropriate the value of future enhancements via price hikes. Once the developers' enhancements become part of the open code base, they are relatively free but a user must still purchase the indispensible complement to receive their value. Thus the original author could act as a monopolist with respect to the value of both his own code as well as the value he did not create.

Resolution: In order to encourage ex ante investment (i.e. before becoming sunk costs) by developers, the author can contractually commit to forgoing real dollar price hikes on the version of common code acquired by developers. This provides some assurance that on expiration of a developer's proprietary period, forward value created by the developer will not simply be expropriated by the author. The author, however, need not commit to price levels on his own future development. This leaves the author free to continue adding value through innovation but, analogous to secondary markets in durable goods, future prices will be conditioned by the presence of an inferior substitute. Prices on author enhancements will be proportional to the marginal value created rather than the growing stock of value created, which is socially more efficient.

Note that if developers do not add sufficient value to make opening the code worthwhile, then the author will not open the source in the first place. This merely restores the relevance of standard licensing arrangements.

V. Conclusions

In this paper, we explore the tension that a platform author faces between fostering platform adoption and follow-on innovation versus immediately capturing the profits to be made from the platform itself. Employing two-sided network externalities, we explore how the release of free information benefits those who develop as well as those who consume. We consider the welfare of consumers and platform authors, and show that environmental parameters such as the size of the consumer population, the developer pool, the magnitude of externalities in each direction, and the ability to reuse code can affect the optimal choice of time to release and degree of openness. One contribution of this research is to demonstrate how better licensing can move a monopolist in the direction of a social planner. We show how such a firm benefits by using intellectual property rights to grow their interest in subsequent innovation. Opening a platform increases the attractiveness of third party investment. Controlling the timing of third party disclosure alters both their incentives and also the availability of new code. Longer disclosure times boost incentives; shorter disclosure times open the code stock. Balancing these forces leads to a limitations in the duration of proprietary incentives.

Our findings differ from important contributions to the economics of intellectual property in that infinitely long and narrow or infinitely short and broad protections (Katz & Shapiro 1985; Klemperer 1990) need not be optimal. This finding resembles that of Green & Scotchmer (1995) in that "standing on the shoulders of giants" is essential to scientific advancement. That which comes after builds on that which comes before. Where others have shown that shorter term property rights reduce first innovator incentives when different firms undertake a sequence of innovations, we develop a mechanism to facilitate this result. This is also achieved with a minimum of negotiation via a default contract. Under traditional closed licenses, a third party developer with a good idea might need to negotiate access to source code. Either through negotiation or by observing the identity of the developer, however, the original firm might discern the idea and appropriate it. In practice, large software firms have been accused of this by smaller software firms (Jackson, 1999). This risk reduces their willingness to invest or disclose the idea ex ante. In contrast, a default offer of σ and t gives developers an option to enter the market for any fixed costs up to the amount they can recover and without current period disclosure to the platform author.

A second contribution is to build a general framework within which to compare licensing archetypes, including the ability to compare licenses based on developer interest in access freedom versus profits. We show that the welfare maximum is achieved under freedom motivated development and that the welfare maximizing choices under profit motivated development depend on network effects and incentives. Further, if network effects are large enough, a for-profit platform author prefers opening the platform to keeping it closed even relative to direct subcontracting with developers. A proprietary incentive period, however, can strictly enhance welfare in the sense that it represents an option required by some but not exercised by others. Interestingly, the optimal contract can increase both user participation and developer profits in future periods by limiting developer ability to charge in the current period. A platform author that internalizes these tradeoffs can realize higher profits and yet will make choices more like those of a social planner than one who fails to account for third party innovation and network effects.

ACKNOWLEDGMENT

This work has been supported by National Science Foundation Grant IIS-0338662. We are thankful to the participants of the 2003 Workshop on Information Systems and Economics, the 2003 Institute for Operations Research and the Management Sciences Annual Meeting, the 2004 Production and Operations Management Society Meeting, and the 2004 Toulouse Conference on Two-Sided Markets. In addition, we thank Emanuel Farhi for his close reading of the manuscript and suggestions to improve model formulation. We thank Lawrence Lessig, Tom Malone, Eric Raymond, Richard Stallman, and Jean Tirole for their discussion about the problem and model formulation. We also thank the participants of the Creative Commons Business Commons discussion thread.

REFERENCES

[1] Armstrong, M. (2002). "Competition in Two-Sided Markets." mimeo: Nuffield College.

[2] Benkler, Y. (2002). "Coase's Penguin or Linux and the Nature of the Firm" *Yale Law Journal* 112(3). pp 369-447.

[3] Bakos, Y. and E. Brynjolfsson (1999). "Bundling Information Goods: Pricing, Profits, and Efficiency," *Management Science* 45(12): 1613-1630.

[4] Caillaud, B. and B. Jullien. "Chicken & Egg: Competing Matchmakers." mimeo: University of Toulouse.

[5] Farrell, J. & Gallini, N. (1988) "Second-sourcing as a commitment: monopoly incentives to attract competition" *Quarterly Journal of Economics* 103(4) pp 673-694

[6] Farrell, J., Monroe, H. & Saloner, G. (1998) "The Vertical Organization of Industry: Systems Competition vs Component Competition" *Journal of Economics & Management Strategy* 7(2) pp 143-82.

[7] Gilbert & Shapiro (1990). "Optimal patent length and breadth" *Rand Journal of Economics* 21(1), 106-112.

[8] Green, J.R. & S. Scotchmer (1995). "On the division of profit in sequential innovation," *Rand Journal of Economics* 26(1), 20-33.

[9] Hippel, E.V. & G.V. Krogh (2003). "Open Source Software and the 'Private-Collective' Innovation Model," *Organization Science* 14(2), 209-223.

[10] Hippel, E.V. (1994). " 'Sticky Information' and the Locus of Problem Solving: Implicaitons for Innovation," *Management Science* 40(4), 429-439.

[11] Jackson, T. P. (1999). "U.S. v. Microsoft: Findings of Fact." United States District Court for the District of Columbia.

[12] Katz, M. L. and C. Shapiro (1985). "Network Externalities, Competition, and Compatibility." *American Economic Review* 75(3): 424-440.

[13] Klemperer, P (1990) "How broad should the scope of patent protection be?" *Rand Journal of Economics* 21(1), 113-130.

[14] Lemley, M. (2004) "Property, Intellectual Property & Free Riding" mimeo: Stanford Law School, WP 291.

[15] Lessig, Larry (2001) "Future of Ideas" Random House: New York.

[16] Parker, G. and M. Van Alstyne (2000). "Information Complements, Substitutes, and Strategic Product Design." Social Sciences Research Network (November 8, 2000) http://ssrn.com/abstract=249585.

[17] Parker, G. and M. Van Alstyne (2002). "Unbundling in the Presence of Network Externalities." Mimeo.

[18] Raymond, Eric (2000) "The Magic Cauldron" http://hurkle.thyrsus.com/~esr/writings/.

[19] Rey, P. and J. Tirole (2003) "A primer on foreclosre" forthcoming in Handbook of Industrial Organization III (eds.) Armstrong, M & Porter, R.

[20] Rochet, J.C. and J. Tirole (2003). "Platform Competition in Two-Sided Markets." Journal of the European Economic Association.

[21] Shapiro, C. (2001) "Navigating the patent thicket: cross-licenses, patent-pools, and standard-setting" in *Innovation Policy and the Economy*, Vol 1. (eds.) A. Jaffe, J. Lerner, S. Stern; MIT Press.

[22] Shapiro, C.; Varian, H. R. (1999). *Information Rules: A Strategic Guide to the Information Economy*. Boston, MA, Harvard Business School Press.

[23] Stallman, R. (1992) "Why Software Should be Free" *Free Software Foundation*, April 24.