Topics

Vidya Atal* and Kameshwari Shankar

Developers' Incentives and Open-Source Software Licensing: GPL vs BSD

Abstract: One of the puzzling aspects of open-source software (OSS) development is its public good nature. Individual developers contribute to developing the software, but do not hold the copyright to appropriate its value. This raises questions regarding motives behind such effort. We provide an integrated model of developers' incentives to describe OSS development and compare restrictive OSS licenses that force all modifications to be kept open with non-restrictive OSS licenses that allow proprietary ownership of modified works. Different incentives govern effort provision at different stages of the software development process. We show that open-source licenses can provide socially valuable software when a proprietary license fails to do so. We also show that restrictive OSS licenses generate greater effort provision in the design stage of software development relative to non-restrictive licenses. Endogenizing licensing choice, we find that a project leader chooses a non-restrictive OSS license if reputational concerns drive developers' incentives, a proprietary license when there is a large population of users in the market and a restrictive OSS license if user population is small but reputational benefit is high. Our results resonate well with empirical findings and suggest additional testable implications about the relationship between licensing and software project characteristics. Finally, we also find that the market under-provides restrictive OSS licenses relative to the efficient level, suggesting the need for subsidizing restrictive licenses in some cases.

Keywords: licensing, open-source software, restrictive license, developers' incentives

DOI 10.1515/bejeap-2014-0007

***Corresponding author: Vidya Atal**, Economics and Finance, Montclair State University, 1 Normal Avenue Partridge Hall 427, Montclair, NJ 07043, USA, E-mail: atalv@mail.montclair.edu

Kameshwari Shankar, Economics and Business, City College of New York, City University of New York, New York, NY, USA, E-mail: kshankar@ccny.cuny.edu

1 Introduction

The development of software under open-source licenses has been the subject of considerable attention among economists in recent years, particularly due to the unique features of open-source software (OSS) development and its increasing importance in software markets that were traditionally dominated by large proprietary sellers. One of the puzzling aspects of OSS development is its public good nature where individual developers contribute to developing the software but do not hold any copyright on their work and hence cannot prevent their contributions from being copied, modified or used by others. In the absence of copyrights, developers cannot appropriate the value of their innovations. This calls into question the motives behind such effort by developers.

Beyond ideological or altruistic motivations that may be driving the open-source movement, the literature on OSS has primarily identified two economic benefits that open source provides to developers. First is a direct user benefit to developers as they improve the quality of a software program for their own use. Contributions driven by this motive have public good characteristics since these improvements can be enjoyed not just by the individual developer, but by all other users of the software as well. A second benefit is labor market signaling or reputational benefits from solving programming problems (Lerner and Tirole 2002). This is a private benefit to the programmer and hence is devoid of any under-provision problem of a public good. Much of the research on OSS has adopted one or the other explanation for incentives to develop OSS. Johnson (2002), Harhoff, Henkel, and von Hippel (2003) and Atal and Shankar (2014) consider OSS as the private provision of a public good where developers begin a project with the expectation of using the software.¹ Other papers such as Blatter and Niedermayer (2008) and Spiegel (2009) provide a model based on labor market signaling.

In this paper, we argue that the value of software created by an open-source model depends crucially on the type of incentives governing effort provision by developers. In particular, effort provision can be more or less than efficient depending on these incentives. We show that if contributions are driven by the user value of the software, the benefits of which cannot be appropriated completely by the developer, then the OSS is characterized by the classic under-provision of effort in a public good. On the other hand, if

¹ The private provision of public goods was modeled in a general setting in Bergstrom, Blume, and Varian (1986).

incentives are driven by reputational concerns and ego gratification, then there is a tendency for developers to over-invest their effort in a race to innovate. We thus provide an integrated model of developers' incentives to describe OSS development.

In our model, different incentives govern effort provision at different stages of OSS development. Software development usually begins with the initial conception of the idea and designing the software followed by testing (alpha and beta testing), ending with a mature or stable release of a final version that is ideally made available to the general market. Thus, we describe the software development process in two stages with the first stage being the design stage followed by commercialization of the software in the second stage. Typically, one can expect that the incentives for effort provision vary depending on the phase of software development. In the initial stages, when the idea is being developed and the preliminary software code is being improved, the reputational or labor market signaling payoff from effort is significant. There are a few reasons for this. First, the software versions being updated are more visible to the software community and hence the signals are targeted toward the relevant labor market for the programmer. Second, the intellectual challenge associated with developing a new idea is likely to be greater; hence, the ego gratification payoff from being the first to innovate is large. On the other hand, later stages of software development are often geared toward making the software more useful to end-users through documentation, adding usability features to the software, and other aspects that make it easier to commercialize the software to the general consumers' market. Effort provision under these two incentives may not be optimal; reputational concerns tend to create over-investment in design effort, while the public good under-provision can produce suboptimal effort levels in both stages.

This framework allows us to describe the role of the OSS licensing process in inducing more optimal effort across different stages of software development. While there is a wide range of open-source licenses each with different terms under which the software can be modified, combined and distributed, in this paper we focus on the restrictions placed on the adoption of a proprietary license by modified works originating from the first open-source license. The most restrictive license in these terms is the GPL (GNU Public License) that forces derivative works to adopt the same "open"-licensing terms. At the same time, there are other less restrictive OSS licenses, such as the BSD (Berkeley Software Distribution) license, that allow modifications to be released under a different license, even a proprietary one.

Our analysis yields some interesting predictions about the motivations driving effort provision among different OSS licenses. First, unlike a

restrictive license (*R*) where the final software is freely available to all users, under a non-restrictive license (*NR*), which can be made proprietary at a later stage, developers know that they may have to pay a price to use the final software in the future. This limits their consumers' surplus as users and hence tends to suppress their design effort below the level achieved by an *R* license in the first stage. This also implies that unlike an *R* license where design effort is driven by both reputational benefits and expected user benefits, effort in the early stages of the *NR* license is driven only by reputational concerns. This makes effort provision under the *NR* license more sensitive to the size of the reputational prize as compared to the *R* license. Thus we find that *R* licenses always provide more effort in the design stage relative to *NR* licenses. This result speaks to the findings by Colazo and Fang (2009) that restrictive licenses are associated with more coding activity relative to non-restrictive ones.

Comparison of the commercialization value generated by the two licenses is more complex. Due to complementarities in the two stages of software development, the difference in commercialization effort between the two types of licenses may go either way. On the one hand, for any given level of design effort, the public good problem in an *R* license leads to under-provision of commercial value in the second stage. This does not happen under an *NR* license which operates as proprietary software in this stage. At the same time, if design and commercialization efforts are complementary, higher design effort may stimulate greater commercialization effort. Since effort in *NR* licenses is more sensitive to reputational benefits, when these are large, design effort under the two licenses is similar, so that the public good problem in the *R* license outweighs complementarity effects leading to an overall reduction in commercial value below that provided by the *NR* license. The opposite is true when reputational gains from software development are small.

In contrast to our finding that *R* licenses may be more successful than the *NR* license in generating the software, much of the empirical research on the success of OSS has found favor with *NR* licenses (Comino, Manenti, and Parisi 2007; Subramaniam, Sen, and Nelson 2009). We address this puzzle in our model by endogenizing the choice of license. By doing so, we find that the market chooses an *NR* license when developers' incentives are driven by reputational concerns. Market selection of *NR* licenses in the presence of significant reputational benefits and consequently greater effort provision in this license then explains, in equilibrium, the finding that *NR* licenses are more successful in generating software for users. Further, we find that the *NR* license dominates the *R* license when there are a large number of users of the software. This is

supported by the finding in Comino, Manenti, and Parisi (2007) that *NR* licenses are more likely to be found in software with larger commercial value.

We also describe effort provision and equilibrium licensing choice when a fully proprietary license is an option. Unlike the OSS licenses, a proprietary license does not provide reputational benefits to developers; however, it provides more optimal user value relative to effort costs. One interesting result here is that when the cost of effort provision is too high, the proprietary license fails to produce the software even though it is socially beneficial, whereas both OSS licenses produce it. This is in contrast to the result in Johnson (2002) where he shows that OSS may not generate socially valuable software that a proprietary production process provides. In terms of licensing choice, we find that a proprietary license is chosen over an OSS license by the market when reputational benefits are small but the user population is very large. Lerner and Tirole (2005) find that non-restrictive licenses dominated OSS that targeted other software developers while restrictive licenses were more likely to be used in software geared toward end-users. We believe that the results of our analysis address this empirical finding since reputational gains are likely to be high when the target audience is comprised of other knowledgeable software developers.

Finally, we also provide a welfare analysis of the licensing choice. We find that an *R* license is under-provided by the market relative to the efficient level. Sometimes the market chooses an *NR* or a proprietary license even when *R* is socially optimal. The market's choice of the license is driven by the surplus to the developer. Since the software developed by the *R* license is never commercially sold for a positive price, the market ignores the end-user surplus generated by this license. The resulting positive externality from the *R* license provides a case for preferential public policy toward such licenses. We also find that a proprietary license chosen by the market may sometimes be desirable from an efficiency point of view. Hence a blanket subsidization of OSS without regard to the project's characteristics is suboptimal.

Our paper draws on two streams of research on OSS. First is the literature on economic incentives for the development of OSS. As mentioned before, much of the existing literature has either adopted a public good provision model or a labor market signaling perspective for developers' incentives. In contrast, we allow for both incentives and describe the interaction between the two incentives in the OSS development process. Second, we also contribute to the research on OSS licenses. Literature in this area is less extensive. Gaudeul (2005) examines the choice between GPL and BSD licenses when developers can hijack the project and sell it for a positive price. The BSD license allows greater effort provision, but also entails the possibility that the project leader loses some

profits to developers. Niedermayer (2013) and D'Antoni and Rossi (2007) also look at licensing choice as a way of balancing out investment incentives across complementary components of software production as we do here. However, while they only look at the hold-up of developers' investment in the software production process, we look at both the hold-up problem and the potential over-investment problem from the tournament to win the reputational prize. We argue that the hold-up of investment may be less severe in some cases because of the mitigating effect of this over-provision.

The paper is organized as follows. Section 2 describes the model. Section 3 describes effort provision under three kinds of licenses – a proprietary license, a restrictive OSS license and a non-restrictive OSS license. Section 4 explains the choice between these three licenses under various conditions. Section 5 describes the welfare maximizing choice of license and compares it to the equilibrium license. Section 6 concludes. All proofs are provided in Appendix B.

2 The Model

We consider a model with a profit maximizing project leader organizing the development of a software. There are $M > 2$ consumers in the population. $N \in (1, M)$ of the consumers are also developers who exert effort to develop the software and then consume the final software product. The remaining $(M - N)$ consumers are end-users who simply consume the software.

Software development occurs in two stages – Stage 1 or the “Design” stage where developers invest effort to design and test the software and Stage 2 or the “Commercialization” stage where developers add user value to the software to make it commercially usable by the general market.²

The software can be produced under three different kinds of licenses – a proprietary license (P), a restrictive OSS license (R) or a non-restrictive OSS license (NR).³ In the P license, the project leader controls effort provision in every stage of the production process and appropriates all the consumers' surplus from the software. In an R OSS license, neither the project leader nor the developers hold the copyright for the value of their effort and the OSS is provided as a public good. The project leader does not control effort provision in any stage in a restrictive OSS license. In an NR OSS license, the project leader does not control the design stage of the software, but she can make the software

² See Reinganum (1989).

³ Restrictive are also called “copyleft” licenses.

proprietary during the commercialization stage and appropriate the resulting consumers' surplus from the software by charging a positive price.

To provide some real-world context to our model of the production process, let us think of the project leader as a software platform owner, such as Apple, Google or a Linux distributor. In the case of Apple, the platform is completely proprietary as Apple controls both the operating system and the application development process. On the other hand, Google has adopted a non-restrictive open-source license (Apache license) for its Android operating system. Thus while Google does not control the platform itself which is open to all developers, it does have its own proprietary applications on the platform (such as its search, maps, mail, etc.). Finally, there are a number of Linux distributors, both commercial (Ubuntu) and community driven (Debian), who use a restrictive open-source license (GPL) and hence allow developers to fully control the platform and the application development process.

Given these three license choices, the project leader chooses a license for her project in Stage 0 in a competitive market for projects and “hires” N developers.⁴ Project leaders, competing to “hire” the best available developers, thus have to pay their entire profits as an entry wage to attract developers.

Effort by developer i in stage t is $e_{it} \in [0, 1]$, where $t \in \{1, 2\}$ and $i \in \{1, 2, \dots, N\}$. The marginal cost of effort for each developer is $c > 0$.⁵ We assume that $c < 2M$. This ensures that there is a positive probability of the software being developed.

In Stage 1, there is a stochastic process that determines how effort translates into design value for each developer in this stage. The design value generated by developer i is $d_i = e_{1i}\varepsilon_i$, where $\varepsilon_i \stackrel{\text{i.i.d.}}{\sim} U[0, 2]$. This means that even if a developer puts a lot of effort, sometimes the design might not be innovative and successful, or sometimes it might add much more design value than the actual effort invested. The highest design value generated by developers, denoted by $D = \max\{d_i | i = 1, 2, \dots, N\}$, proceeds to Stage 2 of the software development process. In Stage 2, developers again exert effort to make the software usable to the end-user. The commercial value of the software generated in this stage is $Z = \sum_{i=1}^N e_{2i}$.

The total value of the software development process to developer i then comprises three components given license type $L \in \{P, R, NR\}$ and effort levels

⁴ In reality, a project leader can start a project under, for example, a proprietary license and then turn it into an OSS license, but we assume away these possibilities in our model.

⁵ We assume that all developers are identical, hence they have the same utility from the software and the same cost of effort.

e_{1i} and e_{2i} . First is the user value, u , from using the end product. We assume a multiplicative utility function for this value implying that design and commercialization values are complementary to each other, i.e.

$$u = DZ.$$

Second, if an OSS license is adopted (R or NR), developers' contributions in the design stage provide a potential reputational benefit from making the innovation. However, only the developer who makes the highest value contribution (d_i) receives this benefit. We assume that developers may receive this reputational benefit only from the design effort and not in the later commercialization stage.⁶ If the license adopted is proprietary, then developers' contributions are not visible since the project leader owns the copyright in all stages of the software production process and hence there is no reputational benefit to developers from investing in design effort.⁷ Thus, we define this reputational prize as $S(L)$, where

$$S(L) = \begin{cases} S > 0 & \text{if } L \in \{R, NR\}, \\ 0 & \text{otherwise.} \end{cases}$$

It is worth elaborating on the nature of this reputational prize, S , in our model. Lerner and Tirole (2002) make a distinction between "career concerns incentive" and "ego gratification" in what they broadly define as the signaling incentive for developers to contribute toward OSS. Under the career concerns motive, developers contribute to OSS because it provides them a way of signaling their ability to future employers. We do not model the future labor market outcomes of developers beyond the current OSS project. Hence our analysis does not capture the true signaling aspects of OSS contributions when employers cannot directly observe developer ability. Thus, in the context of our model, the reputational prize, S , should be interpreted as an ego gratification payoff that increases the

⁶ In our view this is a reasonable assumption given what programmers themselves say about what motivates them. For example, noted open-source advocate Eric Raymond (1999), in his treatise on the development of Linux, says that programmers are driven by ego gratification and building a reputation among hackers. According to him, the open-source process channels developers' ego utility to look for complex errors in the code, which is much more challenging than actually fixing the bugs once they are found. Raymond also acknowledges that programmers do not enjoy documentation which involves describing details about the program's features to other users and application developers.

⁷ It may be argued that proprietary firms can also provide reputational incentives for their developers by publicly providing signals of employee performance, for example through promotions (see Bar-Isaac, Jewitt, and Leaver 2014). However, without making the actual software contributions public, the proprietary license owner can never achieve the same degree of transparency as an open-source license does.

utility of the winning developer. In this sense, the value of this prize is also a part of the total welfare maximized by the social planner.⁸

Third, an OSS license also provides some altruistic utility to the developers, which is denoted by $B(L)$, where⁹

$$B(L) = \begin{cases} B^r & \text{if } L = R, \\ (B^r - \Delta B) & \text{if } L = NR, \\ 0 & \text{otherwise.} \end{cases}$$

We assume that $0 \leq \Delta B \leq B^r$, i.e. the altruistic payoff from a fully open license is higher than the payoff in a license that allows proprietarization. Thus the total value to the developer across all stages of the software development is $V = B(L) + E[S(L)] + E[u]$, where $E[S(L)]$ is the expected value of the reputational prize and $E[u]$ is the total expected user value realized at the end of Stage 2. Finally, note that the $(M - N)$ end users only get the user value from the final software product, $E[u]$.

The timing of the game is as follows. The project leader, facing a competitive market for hiring the N developers, chooses a license and pays an entry wage to attract these developers. After the developers are hired, they provide effort toward the design value of the software. The developers then exert effort to provide commercial value to the software. After the commercialization process, the final software is consumed by the developers themselves and other end-users and the game ends.¹⁰ Following is the description of each stage of the game in summary:

$$\text{Stage 0} - \begin{cases} \text{Decision maker :} & \text{Project leader} \\ \text{Actions :} & \begin{aligned} & \text{(i) Choose a license (proprietary, restrictive,} \\ & \text{or non – restrictive).} \\ & \text{(ii) Compete to “hire” } N \text{ developers by giving} \\ & \text{entry wage.} \end{aligned} \end{cases}$$

8 We briefly discuss the implication of this specification for the signaling payoff in Section 5, where we examine the welfare implications of an open-source license.

9 We can think of this as the “reciprocal altruism” described by Athey and Ellison (2014), although we do not explicitly model this part of the utility as they do.

10 In software parlance, the stage game described here is known as the “waterfall” model where software development occurs in a sequence of non-parallel steps. This has been the traditional model of software development and is widely used by many project managers. More recently, some software experts have been promoting an “iterative” model where development occurs through a series of loops with constant feedback from earlier stages to later stages of the process. Each loop within the iterative model can be thought of as a reduced form waterfall model. In that sense, our model can be easily extended to be a finitely repeated game without substantially altering the main predictions of the paper. We thank an anonymous referee for pointing us to this distinction.

Stage 1	— {	Decision maker :	Project leader if proprietary, developer if OSS
		Actions :	Effort provision in design stage
Stage 2	— {	Decision maker :	Project leader if proprietary or non – restrictive, developer if restrictive.
		Actions :	Effort provision in commercialization stage.

The equilibrium concept used is Subgame Perfect Equilibrium (hence we solve the game using backward induction to find the equilibrium). We restrict our focus to symmetric equilibria only. We solve Stage 2 and then Stage 1 to get the effort provision under each licensing choice in the next section. Then, given these, we solve Stage 0 to determine the equilibrium choice of license in Section 4.

3 Effort Provision by Developers

The incentives for effort provision by developers depend on the stage of the software development process and the type of license under which the software is being produced. Below we describe the equilibrium effort and the resulting software's user value under each of the three kinds of licenses. We then describe the licensing choice in the next section.

3.1 Benchmark: First Best Efficient Outcome

Before we look at the equilibrium effort provision under each license type, it is useful to examine how a welfare maximizing social planner sets effort levels at each stage. Note that, for any given effort level in both stages, the restrictive OSS license provides the highest welfare across all license types. This is because the open development process not only provides a reputational payoff S to the winner by making her innovation visible to the software community, it also provides the highest altruistic benefit to the developers. Hence a social planner maintains openness at all stages of the development process. Given this production process, at each stage, the social planner chooses effort to maximize the total value across all M users net of effort cost across the N developers. So in Stage 2, given the realization of D , the social planner chooses e_2 to maximize:

$$V_2^o(e_2|D) = M(DNe_2) - Nce_2. \quad [1]$$

Let $e_2^o(D)$ represent the optimal effort in Stage 2. Comparing the marginal cost and marginal benefit from commercialization effort, the social planner

chooses positive effort if and only if $D \geq \frac{c}{M}$. In other words, positive commercialization effort is efficient if and only if the realized design value exceeds a threshold. Thus,

$$e_2^o(D) = \begin{cases} 1 & \text{if } D \geq \frac{c}{M}, \\ 0 & \text{if } D < \frac{c}{M}. \end{cases} \quad [2]$$

Since commercialization of the software in Stage 2 is efficient if and only if D is high enough, this means that if the software is developed at all by the social planner, design effort in Stage 1 must be high enough, in particular, $e_1^o \geq \frac{c}{2M}$. If $e_1 < \frac{c}{2M}$, then $e_1 \varepsilon_i < \frac{c}{M}$ for all i , so that $D = \max_i \{e_1 \varepsilon_i\} < \frac{c}{M}$ and $e_2^o = 0$. For all $e_1 \geq \frac{c}{2M}$, the total expected surplus from design effort is:

$$V_1^o(e_1) = NB^r + S + NE[(MD - c)e_2] - Nce_1. \quad [3]$$

Let us define the stochastic variable $x = \max_i \{\varepsilon_i\}$, then $D = e_1 x$. Let us define $f(x)$ as the density function for x over the support $[0, 2]$ which is derived from the i.i.d. uniform distributions of each ε_i for $i = 1, 2, \dots, N$. Then, given e_2^o described in eq. [2], commercialization effort is exerted if and only if $x \geq \frac{c}{Me_1}$, so that the total expected surplus is:

$$V_1^o(e_1) = N \left\{ B^r + \frac{S}{N} + \int_{\frac{c}{Me_1}}^2 (Me_1 x - c) f(x) dx - ce_1 \right\},$$

and the derivative with respect to e_1 is:

$$\frac{\partial}{\partial e_1} V_1^o(e_1) = N \left\{ \int_{\frac{c}{Me_1}}^2 Mx f(x) dx - c \right\}.$$

As we can see above, the marginal benefit from higher design effort is only realized when commercialization occurs, i.e. for $x \geq \frac{c}{Me_1}$. However, the marginal cost of design effort is always just the constant c . As design effort increases, the probability that commercialization occurs increases, so that the expected marginal benefit increases. As a result, the total expected surplus is convex in design effort. Also observe that effort level only influences the user value of the software and not the reputational prize.

Given the convexity of the expected surplus from the software $V_1^o(e_1)$, the optimal design effort is either zero so that the software is not developed at all by the social planner or $e_1^o = 1$. The social planner chooses to produce the software and $e_1^o = 1$ as long as the expected surplus is positive: $V_1^o(1) \geq 0$. When B^r and $\frac{S}{N}$

are very high, this is true at all effort costs. However, if B^r and $\frac{S}{N}$ are low, $V_1^o(1)$ can be negative if the marginal effort cost is high. When this is possible, let us define $\bar{c}^o$ as the threshold level of marginal cost where $V_1^o(1, \bar{c}^o) = 0$. Then the social surplus from the software is positive if and only if $c < \bar{c}^o$.

The following lemma characterizes the first best outcome for effort provision at each stage.

Lemma 1 *Software development is not efficient if the altruistic and signaling benefits are very low and marginal cost of effort is very high, i.e. $B^r + \frac{S}{N} < 2M$ and $c > \bar{c}^o$. Otherwise, it is always efficient to develop the software under an open-source license with $e_1^o = 1$ and $e_2^o(D)$ as defined in eq. [2].*

Lemma 1 shows that expected surplus is negative and software development is not socially valuable when developers' benefits from reputation and altruism are low and effort cost is high. As we show in the next section, a proprietary license may not produce the software even if its social value exceeds the effort cost of developing the software.

3.2 Proprietary License

In a proprietary license, the project leader controls the effort provided by the developer at every stage of the production process (refer to the summary of the game at the end of Section 2). We assume that a developer's reservation wage once she has been employed in a given software project is zero.¹¹ Hence the project leader pays every developer a wage exactly equal to her cost of effort. Thus, once they are hired, developers do not make any surplus in the labor market. In the consumers' market, after the final software has been produced, the project leader acts as a monopolist and charges a price that is equal to the consumer's surplus of each user. So developers do not gain any surplus from using the software either and the project leader appropriates all the gains from the software production process.¹² Another important feature of the proprietary license in our model is that developers' design innovations are not made public and hence there is no reputational payoff.

¹¹ This is true for instance if there are non-compete clauses in the employment contract that prevent the developer from working for another project, or if there is a potential for copyright infringement lawsuits by the project leader if the developer tries to use the current software in another venture.

¹² We assume perfect monitoring so there is no moral hazard problem of workers shirking.

Let us start by looking at Stage 2. Given the realized design value D and a commercial value of Z , the total user value of the software is DZ to each user. Since there are M users, if the project leader charges a price $p = DZ$, the revenue from the software is $M(DZ)$. The cost of effort to each of the N developers from effort level e_{2i} is ce_{2i} which is then the wage paid by the project leader. Hence the total cost of production in the second stage is Nce_{2i} . If the project leader chooses an effort level of e_2 for each developer, her profit is:

$$\pi_2(e_2|D) = M(DNe_2) - Nce_2. \quad [4]$$

The project leader's objective function in Stage 2 shown in eq. [4] is identical to the social planner's objective described in eq. [1]. This means that given D , the proprietary license provides ex post efficient commercialization effort, i.e. $e_2^p(D) = e_2^o(D)$.

Now in Stage 1, given $e_2^p(D)$, the project leader chooses design effort once again to maximize total expected profits. Let e_1 be the effort required by each developer. Then the expected payoff to the project leader is:

$$\pi_1(e_1) = NE[(MD - c)e_2] - Nce_1. \quad [5]$$

Comparing eqs [5] and [3], we see that marginal effect of design effort is the same on expected profits as on the social planner's expected surplus. However, since the proprietary license suppresses both the reputational payoff to the winner and the altruistic benefits from sharing the software across all users, profits are lower than the potential social value of the software. This means that the project leader has a lower incentive to develop the software than the social planner.

Let us define $\bar{c}^p$ such that $\pi_1(1; \bar{c}^p) = 0$, i.e. $E(MD - \bar{c}^p) - \bar{c}^p = 0$ where $\bar{c}^p \in (N, M)$.

Proposition 1 (a) A proprietary license develops the software if and only if the marginal cost of effort is low enough (when $c \leq \bar{c}^p$). In that case, $e_1^p = 1$ and $e_2^p(D) = e_2^o(D)$ as defined in eq. [2].

(b) A proprietary license may not provide the software even though a social planner may find it optimal to develop the software, i.e. $\bar{c}^p > \bar{c}^o$.

Part (a) of the proposition shows that as with the first best efficient outcome, the project leader develops the software under a proprietary license if and only if the cost of effort is low enough. Part (b), however, states that a proprietary license may not provide the software even when there is social value from developing it because, for any given effort cost, proprietary profits are lower than the potential social value of the software. In this case, as we show in the

next two subsections, an OSS license may provide the software. This contrasts with the findings by Johnson (2002) who shows that OSS licenses sometimes fail to provide socially valuable software where proprietary production does. Here we find that the opposite can also happen, i.e. an OSS license may provide the software although the user value does not justify the cost of proprietary production. This occurs because, in our model, the proprietary production process does not provide altruistic and reputational benefits to software developers while an open-source process does. Since these benefits are ignored in Johnson's (2002) analysis, the public good aspect of open-source production always creates an under-supply of software relative to proprietary production in his model.

3.3 Restrictive OSS License

In a restrictive OSS license, the project leader does not control effort provision at any stage (refer to the summary of the game at the end of Section 2) and she does not get positive profits from the software. This is because the R license forces the project leader to keep all software developments open. In this case, developers appropriate their entire user value from the software in Stage 2. Thus market price is trivially zero.¹³ Developers are paid a wage of zero and the project leader makes zero profits.

Apart from the user payoff from the final software, developers also get a reputational payoff of S if they produce the highest value innovation in the design stage and an altruistic or ideological benefit B' from being engaged in a copyleft license. In order to ensure the existence of a symmetric equilibrium, we assume that $B' > \frac{N-2}{N}S$.¹⁴

Again, we start with equilibrium effort provision in the commercialization stage. Given the realization of D in the previous stage and effort of all other developers, $e_{2j} = e_2^j$, $j \neq i$, the payoff from effort e_{2i} to developer i is:

$$V_2^i(e_{2i}) = D((N-1)e_2^j + e_{2i}) - ce_{2i}. \quad [6]$$

We see from eq. [6] that developer i exerts effort for commercialization if and only if $D \geq c$. Since $c > \frac{c}{M}$, for any given realization of D , equilibrium commercialization effort is less than optimal, i.e. $e_2^i(D) < e_2^o(D)$. This is driven by the public good nature of the OSS development process where all developers simultaneously benefit from a single developer's effort.

¹³ This is because we have assumed that, once the software has been produced, distribution costs are zero.

¹⁴ Details for this restriction are described in Appendix B.

Next let us look at the incentives to provide design effort in Stage 1. The expected payoff from design effort to developer i given that all other developers contribute e'_1 is:

$$V_1^r(e_{1i} | e'_1) = B^r + E(S; e_{1i}, e'_1) + E[V_2^r(e'_2)] - ce_{1i}. \quad [7]$$

Before we proceed with the equilibrium analysis, it is useful to explain the race to win the reputational prize from design innovation among developers. Since we restrict our results to a symmetric equilibrium, let us consider developer i 's expected value of winning the reputational prize with effort e_{1i} , given that every other developer chooses $e'_{1j} = e'_1, j \neq i$. Developer i wins the prize if and only if $e_{1i}\varepsilon_i \geq e'_1\varepsilon_j$ for all $j \neq i$. Under a uniform distribution for ε_i , the expected value of the prize is given by¹⁵

$$E(S; e_{1i}, e'_1) = S[\Pr(e_{1i}\varepsilon_i \geq e'_1\varepsilon_j \text{ for all } j \neq i)]$$

$$= S \begin{cases} 0 & \text{if } e_{1i} = 0 \text{ and } e'_1 \geq 0, \\ \int_0^2 \left(\frac{e_{1i}\varepsilon_i}{e'_1} \frac{\varepsilon_i}{2}\right)^{N-1} \frac{d\varepsilon_i}{2} & \text{if } e'_1 > e_{1i} > 0, \\ \int_0^{\frac{2e'_1}{e_{1i}}} \left(\frac{e_{1i}\varepsilon_i}{e'_1} \frac{\varepsilon_i}{2}\right)^{N-1} \frac{d\varepsilon_i}{2} + \int_{\frac{2e'_1}{e_{1i}}}^2 \frac{d\varepsilon_i}{2} & \text{if } e_{1i} \geq e'_1 > 0, \\ 1 & \text{if } e'_1 = 0 \text{ and } e_{1i} > 0. \end{cases}$$

Note that in a symmetric equilibrium, all developers put positive effort because the best response to everyone else not working ($e'_1 = 0$) is to put a positive amount of effort ($e_{1i} > 0$), however, small. This means that the software is always developed under an OSS license.

It is useful to note the following features about the reputational payoff. First, a priori, a symmetric equilibrium in effort provision means that the reputational payoff is independent of effort. Since $e'_{1i} = e'_1 > 0$, the expected payoff described above becomes:

$$E(S) = S \left[\int_0^2 \left(\frac{\varepsilon_i}{2}\right)^{N-1} \frac{d\varepsilon_i}{2} \right] = \frac{S}{N}.$$

So before production begins, effort choice has no impact on this payoff and all OSS licenses yield the same expected reputational payoff. However, once

¹⁵ See Appendix A for a detailed derivation of this expected value of winning the reputational prize. Also see Zabochnik and Bernhardt (2001) for a similar tournament model applied to human capital investment.

production has started under a license, the race to be the highest value contributor makes effort relevant to winning the prize. This provides the mechanism for a possible over-provision of design effort with an OSS license in our model.

Next let us look at the expected user payoff in Stage 2 given design effort in Stage 1. Given that all other developers are contributing $e_{1j} = e_1^r$, $j \neq i$, and developer i is contributing e_{1i} , the expected user value net of effort cost to developer i in Stage 2 is:

$$E(V_2^r | e_1^r, e_{1i}) = E(ND e_2^r - c e_{1i}^r),$$

where $D = \max\{e_1^r \varepsilon_j, e_{1i} \varepsilon_i | j \neq i\}$. Although the user benefit is determined by the highest value innovator, all developers get the user benefit from this innovation unlike the reputational prize which is enjoyed only by the winning developer. This creates a public good under-provision problem in design effort.

The symmetric equilibrium in design effort, e_1^r , is derived by comparing the marginal benefit from design effort $\left[\frac{\partial E(S, e_1^r, e_{1i})}{\partial e_{1i}} + \frac{\partial E(V_2^r | e_1^r, e_{1i})}{\partial e_{1i}} \right]_{e_{1i}=e_1^r}$ to the marginal cost, c . Lemma 2 describes e_1^r and $e_2^r(D)$.

Lemma 2 (a) If $c \leq 2$, then $e_1^r = 1$ and

$$e_2^r(D) = \begin{cases} 1 & \text{if } D \geq c, \\ 0 & \text{if } D < c. \end{cases}$$

(b) If $c > 2$, then $e_1^r = \min\{\frac{N-1}{N} \frac{S}{c}, 1\}$ and $e_2^r = 0$.

Lemma 2 shows that the restrictive OSS license always leads to positive effort provision in the design stage because expected marginal benefit exceeds the marginal cost of effort, c . On the other hand, commercialization occurs under this license only if c is low enough.

Comparing part (b) of the lemma to the first best outcome described in Lemma 1, we see that for costs high enough ($c > \bar{c}^o$), the OSS license over-provides design effort. Comparing effort provision under the R license to the proprietary outcome described in Proposition 1, we find two differences. First, with both commercialization and design efforts, there is a public good under-provision problem as every developer free-rides on the efforts of other developers. But the race to win the reputational prize may cause developers to over-invest in design effort in the open-source license. Design effort thus may be more or less than that in P depending on which effect is stronger. Interestingly, it is possible for socially beneficial software development to occur under the

open-source license even though a proprietary license fails to provide the software. This occurs when $c \in (\bar{c}^p, \bar{c}^o)$.

In Proposition 2, we compare effort provision under R and P licenses.

Proposition 2 (a) *Design effort under the restrictive OSS license (R) is strictly higher than that under the proprietary license (P) if marginal cost of effort is high ($c > \bar{c}^p$) and it is strictly lower than proprietary design effort if $\frac{N-1}{N}S < c \leq \bar{c}^p$. In all remaining cases, design effort is the same under R and P licenses.*

(b) *In Stage 2, given the realized design value D , the R license weakly underprovides commercialization effort compared to the P license or socially optimal effort, i.e. $e_2^r(D) \leq e_2^p(D) = e_2^o(D)$.*

The proposition above highlights the public good problem faced by an open-source license. The user value generated from the software in Stage 2 is enjoyed by all developers and end-users. Since the developer who undertakes effort in the commercialization stage only cares about her own value from her effort, the value generated during this stage is less than optimal. This result is stated in part (b) of the proposition above. However, part (a) shows that design effort may be higher or lower under the OSS license as effort provision in this stage is governed by two different incentives. The reputational incentive from producing the highest value innovation creates a tendency among developers to over-invest. At the same time, as in the commercialization stage, the design value generated by the winning developer is enjoyed by all M users whether they contributed with effort or not. When the value of the reputational prize is low, the over-investment effect in the OSS license is weak, so that overall there is under-provision of design effort.

3.4 Non-Restrictive OSS License

With a non-restrictive OSS license, the project leader keeps the software development process open in the design stage, but can organize proprietary production in the commercialization stage (refer to the summary of the game at the end of Section 2) and charge the monopoly price for the final software.¹⁶ Note that if

¹⁶ A valid concern here is that, since the codes developed in the design stage of the software are open, the project leader may not have monopoly power over the final software product after commercialization. At the same time, it is reasonable to assume that the user value added in the commercialization stage produces at the very least a differentiated product giving the project leader some degree of market power. Moreover, one can also expect that the platform provider

the license allows the project leader to make any part of the software proprietary, she always chooses to do so since she can appropriate the entire value of the software.

Here, the project leader does not control effort provision in the design stage. Developers do not receive a wage in this stage, but they do receive a reputational payoff if they produce the highest value innovation. On the other hand, users do not receive any consumers' surplus in the second stage since the project leader extracts their entire user value through the monopoly price. During the commercialization stage, the project leader pays wages equal to the cost of effort. As with the restrictive OSS license, each developer also receives an altruistic benefit. Since the *NR* license does not keep the software development process open at all stages, the altruistic benefit is smaller, i.e. $(B^r - \Delta B)$, where $\Delta B \in [0, B^r - \frac{N-2}{N}S]$. As with the *R* license, we assume that $B^r \geq (\Delta B + \frac{N-2}{N}S)$ to ensure the existence of a symmetric equilibrium for every parameterization of c and S .

In the commercialization stage, given D , the project leader chooses developers' effort to maximize profits, $\pi_2 = [M(DNe_2) - Nce_2]$. Ex post, this yields the optimal effort level achieved through a proprietary license, i.e. $e_2^{nr}(D) = e_2^p(D) = e_2^o(D)$.

Next, in Stage 1, developers are the decision makers (since the design stage is open) and the only incentive for design effort from developers is the reputational payoff from winning the prize (because all other consumers' surplus is extracted by the project leader after commercialization). The expected payoff from design effort to developer i given that all other developers contribute e_1^{nr} is:

$$V_1^{nr}(e_{1i}|e_1^{nr}) = B^r - \Delta B + E(S; e_{1i}, e_1^{nr}) - ce_{1i}.$$

The symmetric equilibrium for design effort, e_1^{nr} , is derived by comparing the marginal benefit of design effort, i.e. $\left[\frac{\partial E(S, e_1^{nr}, e_{1i})}{\partial e_{1i}} \right]_{e_{1i}=e_1^{nr}}$ to marginal cost, c .

Lemma 3 states the equilibrium outcome in effort provision for each stage.

Lemma 3 a) $e_1^{nr} = \min\{\frac{N-1}{N}\frac{S}{c}, 1\}$.

has an inherent advantage over other developers in commercializing its software applications. For example, Google has closed many of its branded applications by making them proprietary. Although its Android-operating system is open source, there have been very few alternative applications of equal quality generated on this platform giving Google a near monopoly over these applications (see Amadeo 2013).

b) If $\frac{N-1}{N}S \geq \frac{c^2}{2M}$, then

$$e_2^{nr}(D) = \begin{cases} 1 & \text{if } D \geq \frac{c}{M}, \\ 0 & \text{if } D < \frac{c}{M}, \end{cases}$$

and if $\frac{N-1}{N}S \leq \frac{c^2}{2M}$, then $e_2^{nr} = 0$.

Part a) shows that design effort is weakly increasing in S . Further part b) shows that if S is too low so that design effort is low, then the realized value of D is always below the threshold and commercialization never occurs.

Comparing the effort levels described in Lemma 3 to proprietary effort levels, we find the following. As with the R license, it is possible to have both over- and under-provision of design effort. Second, although commercialization effort under the P and NR licenses are ex post equal (i.e. for a given realization of D), the expected value of commercialization effort ex ante may be different. For example, when $c > \bar{c}^p$, ex ante commercialization effort is higher under NR licenses (since it is zero under proprietary production). On the other hand, if $c \leq \bar{c}^p$, but $\frac{N-1}{N}S < c$, then expected commercialization level is lower under an NR license.

Relative to the R license, we see that the NR license always provides lower design effort, i.e. $e_1^{nr} \leq e_1^r$. This is because developers only receive the reputational payoff from design effort in the NR license. Ex post, commercialization effort is always higher under NR , i.e. $e_2^r(D) \leq e_2^{nr}(D)$. However, since design effort is higher under R , this may stimulate greater commercialization. Hence the expected value of commercialization effort may be higher or lower than under an R license.

It is also worth noting that since design effort under the NR license is driven solely by the race to win the reputational prize, design effort depends on S , and in particular, increases with S . On the other hand, when the R license generates the possibility of commercialization, i.e. when $c < 2$, the net user value from the commercial software dominates the decision regarding effort provision, so that $e_1^r = 1$ which is independent of S . This generates a counter-intuitive yet interesting hypothesis about non-restrictive OSS licenses – that effort provision is driven more by reputational concerns than commercial user value under non-restrictive licenses relative to restrictive licenses. This is because developers appropriate more user value from the software under an R license than under an NR license.

The following proposition compares efforts across the three licenses discussed above.

Proposition 3 *Comparing design and commercialization efforts across the three licenses, we find the following: (a) Design effort under a non-restrictive OSS license (NR) may be higher or lower than proprietary design effort, however it is always weakly lower than design effort under a restrictive OSS license (R). (b) The expected value of commercialization effort with an NR license is higher than with proprietary license (P) if and only if $c > \bar{c}^P$ (i.e. when P does not provide the software). (c) Relative to an R license, expected commercialization effort under an NR license is higher if and only if either $c > 2$ (i.e. when R does not provide the software) or reputational benefit is high enough $\left(S \geq \frac{Nc}{(N-1)M}\right)$.*

The result derived in Proposition 3 addresses a few empirical findings on the development of software under OSS licenses. First, our result that design effort is higher under an R license is supported by Colazo and Fang (2009) who find that restrictive OSS licenses have more coding activity and faster development speed than non-restrictive licenses. Second, some researchers have found that restrictive OSS licenses have a negative impact on their performance. For example, Subramaniam, Sen, and Nelson (2009) find that restrictive licenses are less likely to generate successful projects. Similarly, Comino, Manenti, and Parisi (2007) find that software distributed under non-restrictive licenses are more likely to reach a mature and stable release. In contrast to these results, we find that relative effort level between R and NR licenses depends on the stage of the development process as well as the size of the reputational prize. In particular, while contributions in the early stages of the development process are always higher under R licenses, effort to commercialize the software may be higher or lower depending on how important reputational concerns are. This apparent contradiction is driven by endogeneity in licensing choice. As we argue in the next section, much of the negative correlation between software performance and license restrictiveness is driven by other factors that influence both effort provision and choice of license. For example, we show that NR licenses are more likely to be selected when reputational benefit, S , is high. Since effort provision in both stages is weakly increasing in S , NR licenses are selected in projects that generate high user value.

4 Choice of License

Before production starts, at Stage 0, the project leader chooses the license under which she organizes software development. We model this licensing choice in a competitive market of many projects where the leader of each of these projects is

competing to attract N developers. This means that every project leader makes zero profits at this stage. Hence, given a particular license, the entry wage paid to developers is simply the profits to the project leader after commercialization.¹⁷ In case of the restrictive OSS license, the entry wage is simply zero, since developers determine their own effort level at every stage and no profits are made on the final software. With the proprietary license and the non-restrictive OSS license, the project leader generates positive profits from selling the final software. Competition among project leaders to hire developers means that this profit is transferred to developers via the entry wage. Given N identical developers, the entry wage for each developer in this case is simply $\frac{1}{N}$ times the project leader's profit.

Further, given a competitive market, in order to attract developers to her project, the project leader must choose the license that maximizes total expected surplus to developers. The surplus to each developer under a given license, $V^d(L)$, is the sum of the entry wage, the altruistic utility, the expected reputational payoff and consumer's surplus that the developer may receive as a user. Since the wage received in each stage of software development is the same as the cost of effort, it cancels out from the surplus calculation. Hence the total surplus to a developer under each kind of license, denoted by $V^d(L)$, is given by:

$$V^d(L) = \begin{cases} E[(MD - c)e_2^p] - ce_1^p & \text{if } L = P, \\ B^r + \frac{S}{N} + E[(ND - c)e_2^r] - ce_1^r & \text{if } L = R, \\ B^r - \Delta B + \frac{S}{N} + E[(MD - c)e_2^{nr}] - ce_1^{nr} & \text{if } L = NR. \end{cases} \quad [8]$$

It is useful to explain what creates the difference in surplus across licenses. First, OSS licenses provide reputational benefit and altruistic utility which is absent in the proprietary model. This tends to make an OSS license more desirable to developers. However, since developers also consume the final software, they care about the user value of the software as well. Since the proprietary license provides optimal effort to maximize user value (relative to effort cost), it may generate greater value to developers than an OSS license. Finally, it is important to note that, in an R license, the final software is freely provided to all users. So even though developers receive positive consumers' surplus, they do not appropriate any consumers' surplus from the $(M - N)$ end-users who only consume the software free of charge. On the other hand, both the P and NR licenses generate a profit by selling to end-users at a positive price. Since this

¹⁷ See Ghosh and Waldman (2010) for a similar setup on how firms choose wage and promotion contracts when there is competition between firms to hire workers.

profit is transferred to developers through the entry wage, it tends to make the R license less appealing to developers.

Next, we compare the surplus to developers from each kind of license in order to determine which license is chosen in equilibrium. Below we characterize the choice between proprietary and open-source licenses for the case where $c \leq 2$ so that commercialization is possible under all three licenses.¹⁸

We first describe the conditions under which a proprietary license is chosen over an open-source one. The primary benefit that the proprietary license yields to developers' total surplus is that, when the proprietary license provides the software, it optimizes user value for the software. On the negative side, it does not provide any altruistic or reputational benefit to developers. When $c \leq 2$, the proprietary license may dominate both the OSS licenses if the reputational benefit S is small. Conversely, if reputational benefit S is high, then an open-source license dominates a proprietary license.

As explained in Section 3.3, when commercialization is possible under the R license (i.e. $c \leq 2$), design effort $e_1^r = 1$ which is independent of the reputational prize S . On the other hand, e_1^{nr} weakly increases with S . In order to understand how the value to developers is affected by the two open-source licenses, let us look at eq. [8].

The value of the R license increases linearly with S , since design effort is independent of reputational incentives. But the value of the NR license is convex in S . This is because a higher reputational prize reduces reputational surplus by stimulating greater competition among developers to win the innovation prize (i.e. $\frac{d}{dS} [\frac{S}{N} - ce_1^{nr}] < 0$), but it improves the net commercial value generated by the license by inducing greater effort, (i.e. $\frac{d}{dS} E[(MD - c)e_2^{nr}] > 0$). As the reputational benefit S gets larger, the marginal effect on commercial value becomes more dominant. As a result, the effect of S on the overall value provided by the NR license first decreases and then increases. Thus for S high enough, the NR license provides high commercial value to users while also generating a higher surplus to developers compared to the R license as long as the altruistic gains from the latter are not too large.

It is straightforward to see that for any given level of developers (N), as the number of users (M) increases, the lost surplus from end-users in the R becomes greater. Thus as the commercial value of the software increases with M , the NR license becomes more valuable to developers relative to the R license.

In order to simplify the exposition of the trade-offs posed by reputational benefit and commercial value, we assume that the difference in altruistic benefit

¹⁸ For $c > 2$, so that commercialization is not possible under R license, a qualitatively similar dominance relationship holds between P , R and NR licenses as described in Proposition 4.

between the OSS licenses, given by ΔB , is not too low so that there is always a positive measure of S where R dominates NR . Let us denote this altruism threshold by $\hat{\Delta B}$. Then we assume that $\Delta B \geq \hat{\Delta B}$ so that when reputational benefit $S = 0$, $V^d(R) \geq V^d(NR)$.¹⁹

Proposition 4 *Suppose that all three licenses provide commercialization effort (i.e. $c \leq 2$) and the difference in altruistic benefit between the non-restrictive (NR) license and the restrictive (R) license is large (i.e. $\Delta B \geq \hat{\Delta B}$). Then the proprietary license is chosen if and only if reputational benefit (S) is low enough; otherwise one of the OSS licenses is chosen. Among the OSS licenses, NR dominates R if and only if the user population (M) and S are large enough (the cutoffs for M and S are described in the appendix).*

Proposition 4 provides insight about how licenses relate to software project characteristics. Lerner and Tirole (2005) find that software projects with high commercial value are likely to be released under non-restrictive licenses. This is supported by the finding in our model that NR is a more valuable license when the user population M is large. Further, we also find that projects that provide high reputational value are more likely to have NR licenses. This provides support to Lerner and Tirole's (2005) finding that software geared toward other developers is more likely to be developed under a non-restrictive license. Software targeting the developer community provides a knowledgeable audience of software intellectuals and hence is likely to involve very large reputational payoffs from effort. In terms of our analysis, this translates into a large S and hence, by Proposition 4, NR dominates R in equilibrium.

Further, endogenizing the license allows us to explain why non-restrictive licenses are more likely to generate successful software. Since the expected reputational benefit from the two licenses is the same (S/N), the NR license is chosen only if it provides greater user value than R .

Proposition 4 also provides empirically testable hypotheses about the relationship between software licensing and project characteristics. As the results show, among software projects that yield the same altruistic and reputational benefits to the developer, projects with larger user population tend to favor less restrictive licenses. At the same time, if reputational benefits depend on how knowledgeable the end-user population is, then for any given size of the end-user base, a larger proportion of software developers among end-users favors an open-source license over a proprietary one; and further within open-source licenses, such projects are more likely to adopt a non-restrictive license.

¹⁹ The explicit parametric restriction on ΔB is described in the proof of Proposition 4 in Appendix B. The results are qualitatively similar for $\Delta B < \hat{\Delta B}$.

5 Welfare Analysis

The equilibrium choice of license that emerges in the market is driven by developers' surplus. In this section, we look at how a social planner chooses the appropriate license to balance out user value to consumers and reputational benefits to developers. The social planner's choice of license then characterizes the second best efficient outcome where effort levels are allowed to be determined by the market. Given the market equilibrium outcome in Stages 1 and 2, the social planner chooses the license to maximize the total surplus to users, developers and the project leader.

Let us denote $V^o(L)$ as the total surplus from license L . $V^o(L)$ is then the sum of the project leader's payoff, developers' surplus and end-user consumers' surplus from the final software. If the license is proprietary or non-restrictive, the surplus of all N developers, $NV^d(L)$, also captures the total surplus for the social planner. This is because, after commercialization, the project leader appropriates the entire consumers' surplus through price and the resulting profits are then transferred to developers through an entry wage at the time of hire. As a result, the social planner's choice between the P and NR licenses is the same as a developer's choice between the two at the time of hire.

However, in the R license, the total surplus is greater than the surplus to all N developers, i.e. $V^o(R) > NV^d(R)$. This is because, in the absence of a positive price for the final software, developers only care about their own user value, but not the user value provided to the $(M - N)$ end-users in the market. Since there is no means of transferring end-user surplus to developers under this license, a positive externality is generated which causes the social value of production to exceed the market's value under R . Total surplus is then given as follows:

$$V^o(L) = \begin{cases} NV^d(L) & \text{if } L \in \{P, NR\}, \\ NB^r + S + NE[(MD - c)e_2^r] - Nce_1^r & \text{if } L = R. \end{cases}$$

The presence of positive externalities in the development of software under the R license implies that the social planner may choose this license even though the market does not. It is also possible that the market's licensing choice sometimes is welfare maximizing and, in particular, that the P license is efficient in some cases. Proposition 5 outlines this result.

Proposition 5 *Retaining the restrictions that $c \leq 2$ and $\Delta B \geq \Delta \hat{B}$, the following describes the features of the second best efficient license chosen by a social planner.*

a) *The proprietary license (P) is efficient if reputational benefit to developers (S) is low enough. In every case where the social planner chooses an OSS license,*

the non-restrictive (NR) license is more efficient than the restrictive license (R) if and only if both the number of users (M) and S are large enough.

b) Comparing the equilibrium and efficient license, the market chooses R licenses less often and the P and NR licenses at least as often as the social planner, i.e. the range of S and M where R (NR or P) is efficient is strictly larger (weakly smaller) than the parameter range where it is chosen in equilibrium.

Proposition 5 allows us to look at the implication of public policies around the choice of license. Governments in many countries including the USA have tried to encourage the use of open-source licenses such as GPL. For example, the UK recently mandated the use of OSS in all government projects whenever available. Similarly, the city of Munich in Germany migrated from Microsoft operating system to OSS and France has also sought to ban the use of proprietary software in government bodies. In the USA, the government provides R&D support for projects released under GPL.²⁰ Part b) of Proposition 5 states that the market under-provides the R license relative to its social value, and hence this provides a justification for preferential public policy toward restrictive OSS licenses. At the same time, it also highlights the fact that the specific nature of the OSS alternative is an important policy consideration. Specifically, we find that, as long as the labor market for developers is competitive, there is no case for preferential treatment in public policy toward NR OSS licenses. Further, as part a) of the proposition shows, it is also possible in some cases that proprietary production chosen by the market is desirable from an efficiency perspective so that a blanket subsidization of OSS with no regard to the project's characteristics is suboptimal.

The results presented in Proposition 5 identify one potential market failure that arises in the choice of licenses, namely positive externalities in the development of OSS under a restrictive license. At the same time we acknowledge that there may be other elements of efficiency that our current model overlooks. For instance, as mentioned in Section 2, we assume that the signaling incentive for effort provision under OSS is driven purely by ego gratification which directly enhances the utility to developers. All else equal, this makes OSS always more efficient than proprietary production. A different reward mechanism for this payoff under OSS may create inefficiencies that are not present in the proprietary model. As an example we could have considered a labor market signaling model like Spence (1973), where winning the design value in an OSS serves as a signal for unobserved worker ability to future employers. This enhances future

²⁰ See Schmidt and Schnitzer (2003) for other examples of countries that have adopted a preferential approach toward OSS.

wages to the developer. If developer effort toward this reward does not add any social value (either in terms of the future productivity of the developer or his utility), then a social planner will have no reason to strictly prefer an OSS license to a proprietary one in the first best outcome.²¹ In reality, the signaling motive for the developer's participation in OSS is likely to be a combination of ego-gratification and labor market signaling. To the extent that developers receive a direct utility payoff from having their innovations publicly observed in an open-source model, our welfare results hold.

6 Conclusion

Our paper provides an integrated theoretical framework to analyze the economic incentives governing various stages of OSS development. We use this framework to explain the choice between restrictive and non-restrictive licenses adopted by various OSS.

We build our theory of OSS development by characterizing a typical software development process as comprising of two broad stages – Stage 1 or the “Design” stage where developers invest effort to design and test the software and Stage 2 or the “Commercialization” stage where developers add user value to the software to make it more commercially usable by a general market. We then describe the incentives governing the investment of effort in each stage of OSS development. Effort provision in the design stage is driven by two motives – reputational benefits from winning a prize to the highest value innovation, and expected future benefits as a user of the OSS. Effort in the commercialization stage is driven solely by user benefits. We then derive the equilibrium effort provision in each stage under a restrictive and a non-restrictive OSS license separately. Finally, we characterize the project leader's choice of license in the context of a competitive market of projects for hiring developers given the equilibrium effort provision following the licensing choice. This allows us to then describe the conditions governing the choice between the three kinds of licenses.

The results of our theoretical model explain empirical evidence about the relationship between features of the OSS and the license adopted for the software. In particular, we explain the following facts about restrictive and non-

²¹ Also see Holmstrom (1999). Holmstrom argues that managers driven by future wage concerns may not make correct investment decisions if they are risk averse and the returns to their investment provide a noisy signal of their true ability.

restrictive OSS licenses. First, software projects released under restrictive licenses are less likely to achieve a mature or stable release (Comino, Manenti, and Parisi 2007). Second, software projects with high commercial value are more likely to be released under non-restrictive licenses (Lerner and Tirole 2005).

Our analysis suggests some additional testable implications. We find that if the number of software users is large, then design effort exerted by developers under OSS licenses is higher than that in a proprietary license. Also, while analyzing the equilibrium choice of license, we find that if reputational benefit is high, then the non-restrictive OSS license emerges in equilibrium. Conversely, when reputational benefit is very low, then a proprietary license is chosen. Among OSS licenses, the restrictive license is chosen when the number of software users is small and reputational gains are low.

From a welfare perspective, we find that the market equilibrium generates fewer restrictive licenses than what is efficient for the society, thus providing a case for some subsidization of restrictive open-source licenses. At the same time, we find that there is no case for subsidizing non-restrictive open-source licenses.

Our model has some limitations. In our model, the number of developers (N) is exogenous. However, in reality, project leaders can decide who joins the project and how many developers should participate in its development. Extending our model to make the number of developers endogenous is likely to provide some interesting implications of the software license on the size of the developer community. In addition, while our paper assumes that a single license is chosen for each software, in some cases software is simultaneously released under a proprietary license as well as an open-source license (dual license). Typically, use of the proprietary license requires a payment to the license owner, while the open-source version requires all modifications to be kept open as with the restrictive license. The two main benefits of using a dual license is to allow the license owner to appropriate some of the future proprietary profits generated from the dual licensed software and to allow license compatibility with other software released under restrictive terms (Valimaki 2003; Comino and Manenti 2011). On the other hand, the restrictive license poses a competitive threat to the proprietary version especially if it generates a better software product to users. Since we do not explicitly model competition in the software market post-commercialization, analyzing the merits of dual licensing is beyond the scope of our current paper. Nevertheless, the model presented here provides fruitful avenues for future extensions to study when and why dual licenses exist. Finally, our paper ignores license compatibility issues that may affect the design or commercialization value of the software. The terms of restrictive licenses, such as GPL, usually create a conflict when the software is combined with other differently licensed code. Thus, if a significant proportion of software is released

under a GPL license, the value of the GPL license for a new software increases as the proportion of other software released under this license increase. This creates a form of network effect in the use of restrictive licenses as more licenses are released in order to maintain this compatibility. We believe that incorporating this issue in examining licensing choice is likely to address many empirical findings such as why a significant portion of open-source code has a GPL license despite the fact that it appears to be less successful than other non-restrictive licenses.

Appendix

A Derivation: Expected Value of the Reputational Prize

Below we derive the expression of $E(S; e_{li}, e_1^*)$, which shows the expected value of winning the reputational prize from producing the highest value innovation in the design stage in an OSS.

Let us consider developer i 's effort, e_{li} , given that all other developers $j \neq i$ invest effort e_1^* . Then developer i gets the reputational prize S if and only if $e_{li}\varepsilon_i \geq e_1^*\varepsilon_j$ for all $j \neq i$, i.e.

$$E(S; e_{li}, e_1^*) = S[\Pr(e_{li}\varepsilon_i \geq e_1^*\varepsilon_j \text{ for all } j \neq i)].$$

If $e_{li} = 0$, then there is no chance of winning the prize, so $E(S; e_{li}, e_1^*) = 0$. At the other extreme, if $e_{li} > 0$ and $e_1^* = 0$, then i wins the prize for sure and $E(S; e_{li}, e_1^*) = S$.

If $e_1^* > 0$ and $e_{li} > 0$, then we have to consider two cases, one where $e_1^* \leq e_{li}$ and the other where $e_1^* > e_{li}$.

- (i) If $e_1^* \leq e_{li}$, then i wins the prize if and only if $\varepsilon_j \leq \frac{e_{li}}{e_1^*}\varepsilon_i$ for every $j \neq i$. Since the support for ε_j is $[0, 2]$, if $\frac{e_{li}}{e_1^*}\varepsilon_i > 2$, i.e. $\varepsilon_i > \frac{2e_1^*}{e_{li}}$, then i wins the prize for sure. For $\varepsilon_i \in [0, \frac{2e_1^*}{e_{li}}]$, i 's probability of winning is $\left[\Pr(\varepsilon_j \leq \frac{e_{li}}{e_1^*}\varepsilon_i)\right]^{N-1} = \left(\frac{e_{li}\varepsilon_i}{e_1^*2}\right)^{N-1}$ under the i.i.d. uniform distribution for ε_j . So the total expected probability of winning over all ε_i is:

$$\Pr(e_{li}\varepsilon_i \geq e_1^*\varepsilon_j \text{ for all } j \neq i) = \int_0^{\frac{2e_1^*}{e_{li}}} \left(\frac{e_{li}\varepsilon_i}{e_1^*2}\right)^{N-1} \frac{d\varepsilon_i}{2} + \int_{\frac{2e_1^*}{e_{li}}}^2 \frac{d\varepsilon_i}{2}.$$

- (ii) If $e_1^* > e_{li}$, then $\varepsilon_i \leq 2 < \frac{2e_1^*}{e_{li}}$, which means that for any given realization of ε_i , the probability of winning S is $\left(\frac{e_{li} \varepsilon_i}{e_1^* 2}\right)^{N-1}$. So the probability of winning the prize in this case is:

$$\Pr(e_{li}\varepsilon_i \geq e_1^*\varepsilon_j \text{ for all } j \neq i) = \int_0^2 \left(\frac{e_{li} \varepsilon_i}{e_1^* 2}\right)^{N-1} \frac{d\varepsilon_i}{2}.$$

Combining the derivations above, the expected value of the reputational prize for developer i , given that she exerts effort e_{li} and all other developers $j \neq i$ exert effort e_1^* , is:

$$\begin{aligned} E(S; e_{li}, e_1^*) &= S \times \text{probability of winning the prize} \\ &= S[\Pr(e_{li}\varepsilon_i \geq e_1^*\varepsilon_j \text{ for all } j \neq i)] \\ &= S \begin{cases} 0 & \text{if } e_{li} = 0 \text{ and } e_1^* \geq 0, \\ \int_0^2 \left(\frac{e_{li} \varepsilon_i}{e_1^* 2}\right)^{N-1} \frac{d\varepsilon_i}{2} & \text{if } e_1^* > e_{li} > 0, \\ \int_0^{\frac{2e_1^*}{e_{li}}} \left(\frac{e_{li} \varepsilon_i}{e_1^* 2}\right)^{N-1} \frac{d\varepsilon_i}{2} + \int_{\frac{2e_1^*}{e_{li}}}^2 \frac{d\varepsilon_i}{2} & \text{if } e_{li} \geq e_1^* > 0, \\ 1 & \text{if } e_1^* = 0 \text{ and } e_{li} > 0. \end{cases} \end{aligned}$$

B Proofs of the Propositions

Proof of Lemma 1. Taking Stage 2 efforts first, given D , $\frac{\partial}{\partial e_2} [V_2^o(e_2|D)] = M(DN) - Nc \geq 0$ if and only if $D \geq \frac{c}{M}$. Hence $e_2^o(D) = 1$ if $D \geq \frac{c}{M}$ and 0 otherwise. In Stage 1, the expected surplus from design effort, $V_1^o(e_1)$ is convex in e_1 . Since the value from not developing the software is zero, the social planner chooses to develop the software as an open source if and only if $V_1^o(1) \geq 0$. $V_1^o(1; c)$ is decreasing in c and it is positive at $c = 0$. If $B^r + \frac{S}{N} - 2M \geq 0$, then $V_1^o(1) \geq 0$ for all $c \leq 2M$. If $B^r + \frac{S}{N} - 2M < 0$, then $V_1^o(1) \geq 0$ if and only if $c \leq \bar{c}^o$, where $\bar{c}^o$ solves $V_1^o(1; \bar{c}^o) = 0$. ■

Proof of Proposition 1. (a) Since $\pi_2(e_2|D) = V_2^o(e_2|D)$, $e_2^o(D) = e_2^p(D)$. In Stage 1, $\pi_1(e_1) = V_1^o(e_1) - NB^r - S$. Hence $\pi_1(e_1)$ is also convex in e_1 , and $e_1^p = 1$ if $\pi_1(1) \geq 0$ and zero otherwise. $\pi_1(1)$ is decreasing in c . It is positive at $c = N$ and negative at $c = M$. Hence we define $\bar{c}^p \in (N, M)$ as $\pi_1(1; \bar{c}^p) = 0$. Then $e_1^p = 1$ if $c \leq \bar{c}^p$ and $e_1^p = 0$ if $c > \bar{c}^p$.

(b) From Lemma 1, if $B^r + \frac{S}{N} - 2M \geq 0$, then we can define $\bar{c}^0 = 2M$ since the social planner always chooses to develop the software and hence $\bar{c}^p < \bar{c}^0$. If $B^r + \frac{S}{N} - 2M < 0$, then at $\bar{c}^p$, $V_1^0(1; \bar{c}^p) = NB^r + S > 0$. Hence again, $\bar{c}^p < \bar{c}^0$. ■

Proof of Lemma 2. First let us derive Stage 2 commercialization effort given the realized value of D . From (6), we can see that $\frac{\partial}{\partial e_{2i}} V_{2i}^r(e_{2i}) \geq 0$ if and only if $D \geq c$. Hence for all i , $e_{2i}(D) = e_{2i}^r(D) = 1$ if $D \geq c$ and 0 otherwise.

Now for the first-stage effort, we need to derive the $E(V_2^r|e_1^r, e_{1i})$. Suppose $G(\cdot)$ denotes the c.d.f. for $D = \max\{e_1^r \varepsilon_j, e_{1i} \varepsilon_i | j \neq i\}$, and $g(\cdot)$ is the corresponding p.d.f. over the support $[0, 2]$. Let us look at the following cases.

(1) If $e_{1i} < \frac{c}{2}$ and $e_1^r < \frac{c}{2}$, then for every realization of ε_i and $\varepsilon_{j \neq i}$, $e_{2i}^r = 0$ since $D < c$. In this case $E(V_2^r|e_1^r, e_{1i}) = 0$.

(2) If $e_1^r < \frac{c}{2} \leq e_{1i}$, then for $x > 2e_{1i} > c$, $G(x) = \Pr(D < x) = 1$ and $g(x) = 0$.

For $x \in [0, 2e_1^r]$, $G(x) = \Pr(D < x) = \Pr(e_1^r \varepsilon_j < x | j \neq i) \Pr(e_{1i} \varepsilon_i < x) = \left(\frac{x}{2e_1^r}\right)^{N-1}$
 $\left(\frac{x}{2e_{1i}}\right)$ and $g(x) = N \frac{1}{2e_{1i}} \left(\frac{x}{2e_1^r}\right)^{N-1} = g_1(x)$, say. For $x \in [2e_1^r, 2e_{1i}]$, $G(x) = \Pr(e_{1i} \varepsilon_i < x) = \frac{x}{2e_{1i}}$ and $g(x) = \frac{1}{2e_{1i}} = g_2(x)$, say. Hence,

$$g(x) = \begin{cases} g_1(x) & \text{if } x \leq 2e_1^r, \\ g_2(x) & \text{if } 2e_1^r < x \leq 2e_{1i}, \\ 0 & \text{otherwise.} \end{cases}$$

Given that $e_{2i}^r = 1$ if and only if $D \geq c$ and 0 otherwise, V_2^r is positive only for $x \in [c, 2e_{1i}]$. Since $c > 2e_1^r$, this means that $E(V_2^r|e_1^r, e_{1i}) = E(NDe_{2i}^r - ce_{2i}^r) = \int_c^{2e_{1i}} (Nx - c)g_2(x)dx$.

(3) If $\frac{c}{2} \leq e_1^r \leq e_{1i}$, $g(x)$ is the same as above, but now $c \leq 2e_1^r$. So

$$E(V_2^r|e_1^r, e_{1i}) = \int_c^{2e_1^r} (Nx - c)g_1(x)dx + \int_{2e_1^r}^{2e_{1i}} (Nx - c)g_2(x)dx.$$

(4) If $\frac{c}{2} \leq e_{1i} < e_1^r$, then for $x \in [0, 2e_{1i}]$ we have $g(x) = g_1(x)$. For $x \in [2e_{1i}, 2e_1^r]$, $G(x) = \Pr(D < x) = \Pr(e_1^r \varepsilon_j < x | j \neq i) = \left(\frac{x}{2e_1^r}\right)^{N-1}$ and $g(x) = (N-1) \frac{1}{2e_1^r} \left(\frac{x}{2e_1^r}\right)^{N-2} = g_3(x)$, say. For $x > 2e_1^r$, $G(x) = 1$. Hence,

$$g(x) = \begin{cases} g_1(x) & \text{if } x \leq 2e_{1i}, \\ g_3(x) & \text{if } 2e_{1i} < x \leq 2e_1^r, \\ 0 & \text{otherwise.} \end{cases}$$

Thus $E(V_2^r|e_1^r, e_{1i}) = \int_c^{2e_{1i}} (Nx - c)g_1(x)dx + \int_{2e_{1i}}^{2e_1^r} (Nx - c)g_3(x)dx$.

- (5) If $e_{li} < \frac{c}{2} \leq e_1^r$, then $g(x)$ is the same as in case (4). However, $e_2 = 0$ for $x \leq 2e_{li} < c$. Hence $E(V_2^r | e_1^r, e_{li}) = \int_c^{2e_1^r} (Nx - c)g_3(x)dx$.

Combining the expected reputational payoff and expected Stage 2 payoff to developer i and after some extensive algebra, the marginal benefit from design effort, $\left(\frac{\partial E(S, e_1^r, e_{li})}{\partial e_{li}} + \frac{\partial E(V_2^r | e_1^r, e_{li})}{\partial e_{li}} \right)$, is:

$$\left\{ \begin{array}{ll} \frac{N-1}{N} \frac{S}{e_1^r} \left(\frac{e_{li}}{e_1^r} \right)^{N-2} & \text{if } e_{li} < e_1^r \text{ and } e_{li} < \frac{c}{2}, \\ \frac{N-1}{N} \frac{S}{e_1^r} \left(\frac{e_{li}}{e_1^r} \right)^{N-2} + 2 \left(\frac{N^2}{N+1} - 1 \right) \left(\frac{c}{2e_{li}} \right)^2 \left(\frac{c}{2e_1^r} \right)^{N-1} + \frac{2N}{N+1} \left(\frac{e_{li}}{e_1^r} \right)^{N-1} & \text{if } \frac{c}{2} \leq e_{li} < e_1^r, \\ \frac{N-1}{N} \frac{S}{e_1^r} \left(\frac{e_1^r}{e_{li}} \right)^2 + 2 \left(\frac{N^2}{N+1} - 1 \right) \left(\frac{c}{2e_{li}} \right)^2 \left(\frac{c}{2e_1^r} \right)^{N-1} + N \left\{ 1 - \frac{N-1}{N+1} \left(\frac{e_1^r}{e_{li}} \right)^2 \right\} & \text{if } \frac{c}{2} \leq e_1^r \leq e_{li}, \\ \frac{N-1}{N} \frac{S}{e_1^r} \left(\frac{e_1^r}{e_{li}} \right)^2 + N + (N-2) \left(\frac{c}{2e_{li}} \right)^2 & \text{if } e_1^r < \frac{c}{2} \leq e_{li}, \\ \frac{N-1}{N} \frac{S}{e_1^r} \left(\frac{e_1^r}{e_{li}} \right)^2 & \text{if } e_1^r \leq e_{li} < \frac{c}{2}. \end{array} \right.$$

Now suppose $c > 2$. In this case, $e_1^r < \frac{c}{2}$ and $e_{li} < \frac{c}{2}$. Then, from the design effort provided by the developers, there is no Stage 2 payoff and the only benefit is the reputational benefit. Here developer i 's best response is to choose $e_{li} \geq e_1^r$. Hence, in a symmetric equilibrium, $e_1^r = \frac{N-1}{N} \frac{S}{c}$ if $\max\{2, \frac{N-1}{N} S\} < c$ and $e_1^r = 1$ if $2 < c \leq \frac{N-1}{N} S$. Under our assumption that $B^r > \frac{N-2}{N} S$, in every case, the value from design effort in equilibrium is positive. Since the threshold c is never met in Stage 2, commercialization is never undertaken and $e_2^r = 0$.

If $c < 2$, then $e_1^r \geq \frac{c}{2}$ in a symmetric equilibrium. This is because if $e_1^r < \frac{c}{2}$ for all $j \neq i$, then i 's best response is $e_{li} \geq \frac{c}{2}$ which cannot be part of a symmetric equilibrium. Hence $e_1^r \geq \frac{c}{2}$ and $e_{li} \geq \frac{c}{2}$. The marginal benefit in a symmetric equilibrium is then $\left[\frac{N-1}{N} \frac{S}{e_1^r} + 2 \left(\frac{N^2}{N+1} - 1 \right) \left(\frac{c}{2e_1^r} \right)^{N+1} + \frac{2N}{N+1} \right]$. It can be checked that, at $e_1^r = 1$, $\left[\frac{N-1}{N} S + 2 \left(\frac{N^2}{N+1} - 1 \right) \left(\frac{c}{2} \right)^{N+1} + \frac{2N}{N+1} \right] > c$. This means that $e_1^r = 1$. ■

Proof of Proposition 2. (a) Comparing e_1^p and e_1^r obtained in Proposition 1 and Lemma 2 respectively, we see that if $c > \bar{c}^p$, $e_1^p = 0 < e_1^r$. For $c \leq \bar{c}^p$, $e_1^p = 1 > e_1^r$ for $\frac{N-1}{N} S < c$ and $e_1^r = 1 = e_1^p$ for $\frac{N-1}{N} S \geq c$.

(b) For $D < \frac{c}{M} < c$, $e_2^r(D) = e_2^p(D) = 0$. If $\frac{c}{M} \leq D < c$, $e_2^p(D) = 1 > e_2^r(D) = 0$. And for $D \geq c$, $e_2^r(D) = e_2^p(D) = 1$. Hence $e_2^r(D) \leq e_2^p(D)$ for all D . ■

Proof of Lemma 3. a) and b) Since in Stage 2, the objective function and decision making process is the same under a non-restrictive OSS license as a proprietary license, $e_2^p(D) = e_2^{nr}(D)$.

In Stage 1, the total value to the developer from design effort is $V_1^{nr}(e_{1i}|e_1^{nr}) = B' - \Delta B + E(S; e_{1i}, e_1^{nr}) - ce_{1i}$. Hence the marginal benefit is simply the marginal reputational payoff, i.e. $\frac{N-1}{N} \frac{S}{e_1^{nr}} \left(\frac{e_{1i}}{e_1^{nr}}\right)^{N-2}$ if $e_{1i} < e_1^{nr}$ and $\frac{N-1}{N} \frac{S}{e_1^{nr}} \left(\frac{e_{1i}}{e_1^{nr}}\right)^2$ if $e_1^{nr} \leq e_{1i}$. Setting $e_{1i} = e_1^{nr}$ in the symmetric equilibrium, we get if $\frac{N-1}{N} S \geq c$, then $e_1^{nr} = 1$ and if $\frac{N-1}{N} S < c$, then $e_1^{nr} = \frac{N-1}{N} \frac{S}{c}$. Under our assumption that $B' > \Delta B + \frac{N-2}{N} S$, the value from design effort is always positive in the symmetric equilibrium. If $\frac{N-1}{N} S \leq \frac{c^2}{2M}$, then $e_1^{nr} = \frac{N-1}{N} \frac{S}{c} < \frac{c}{2M}$, so that $D < \frac{c}{M}$ for every realization of ε_i for all i . This means that $e_2^{nr} = 0$. ■

Proof of Proposition 3. (a) First comparing e_1^p to e_1^{nr} from Proposition 1 and Lemma 3, for $c > \bar{c}^p$, $e_1^p = 0$ while, $e_1^{nr} > 0$, so that $e_1^{nr} > e_1^p$. For $c \leq \bar{c}^p$, $e_1^{nr} \leq e_1^p = 1$. Comparing design effort in the R and NR licenses from Lemmas 2 and 3, for $c > 2$, $e_1^r = e_1^{nr}$. For $c < 2$, $e_1^r \leq e_1^{nr} = 1$. Hence for all c , $e_1^r \geq e_1^{nr}$.

(b) Since $e_2^p(D) = e_2^{nr}(D)$, and the expected value of commercialization effort is weakly increasing in design effort, we have $E[e_2^p(D)|e_1^p] \geq E[e_2^{nr}(D)|e_1^{nr}]$ if and only if $e_1^p \geq e_1^{nr}$. From part (a), this occurs if and only if $c \leq \bar{c}^p$.

(c) For $c > 2$, $e_1^{nr} = e_1^r$ and $e_2^{nr}(D) \geq e_2^r(D)$, hence $E(e_2^{nr}(D)|e_1^{nr}) \geq E(e_2^r(D)|e_1^r)$. For $c \leq 2$, $e_1^r = 1$ and $E(e_2^r(D)|e_1^r) = \Pr(\max\{\varepsilon_i\} \geq c) = 1 - \left(\frac{c}{2}\right)^N$. If $S \leq \frac{Nc^2}{2M(N-1)}$, then $e_2^{nr} = 0$ so that $E(e_2^{nr}(D)|e_1^{nr}) \leq E(e_2^r(D)|e_1^r)$. For $S > \frac{Nc}{N-1}$, $e_1^{nr} = e_1^r = 1$; hence, $E(e_2^{nr}(D)|e_1^{nr}) > E(e_2^r(D)|e_1^r)$. Finally for $\frac{Nc^2}{2M(N-1)} < S \leq \frac{Nc}{N-1}$, $e_1^{nr} = \frac{(N-1)S}{N}$ and $E(e_2^{nr}(D)|e_1^{nr}) = \Pr\left(\max\{\varepsilon_i\} \geq \frac{c}{Me_1^{nr}}\right) = 1 - \left(\frac{c}{2Me_1^{nr}}\right)^N = 1 - \left(\frac{Nc^2}{2M(N-1)S}\right)^N$. Hence, $E(e_2^{nr}(D)|e_1^{nr}) \geq E(e_2^r(D)|e_1^r)$ if and only if $S \geq \frac{Nc}{M(N-1)}$. ■

Proof of Proposition 4. Since $\bar{c}^p > N \geq 2$, for $c \leq 2$, we have:

$$\begin{aligned} & V^d(R) - V^d(P) \\ &= B' + \frac{S}{N} + \frac{2M}{N+1} \left\{ 1 - \left(\frac{c}{2M}\right)^{N+1} \right\} - 2M \left(1 - \frac{c}{2M}\right) + 2 \left(\frac{N^2}{N+1} - 1\right) \\ & \quad \left\{ 1 - \left(\frac{c}{2}\right)^{N+1} \right\} + (2-c) \\ &= B' + \frac{S}{N} - \frac{2N}{N+1} \left[(M-N) + \left\{ (N-1) - \frac{1}{N} \left(1 - \frac{1}{M^N}\right) \right\} \left(\frac{c}{2}\right)^{N+1} \right], \end{aligned}$$

$$\begin{aligned}
& V^d(R) - V^d(NR) \\
&= \Delta B + E[(ND - c)e_2^r] - E[(MD - c)e_2^{nr}] - c(1 - e_1^{nr}) \\
&= \left[\begin{aligned} & \Delta B + 2\left(\frac{N^2}{N+1} - 1\right) \left\{ 1 - \left(\frac{c}{2}\right)^{N+1} \right\} + (2 - c) - c(1 - e_1^{nr}) + \\ & + \begin{cases} 0 & \text{if } S \leq \frac{Nc^2}{2M(N-1)}, \\ \frac{2}{N+1}Me_1^{nr} \left\{ 1 - \left(\frac{c}{2Me_1^{nr}}\right)^{N+1} \right\} - 2Me_1^{nr} \left\{ 1 - \left(\frac{c}{2Me_1^{nr}}\right) \right\} & \text{if } S > \frac{Nc^2}{2M(N-1)}, \end{cases} \end{aligned} \right]
\end{aligned}$$

and

$$\begin{aligned}
& V^d(NR) - V^d(P) \\
&= \left[\begin{aligned} & B^r + \frac{S}{N} + \frac{2M}{N+1} \left\{ 1 - \left(\frac{c}{2M}\right)^{N+1} \right\} - 2M \left(1 - \frac{c}{2M} \right) - \Delta B + c(1 - e_1^{nr}) \\ & + \begin{cases} 0 & \text{if } S \leq \frac{Nc^2}{2M(N-1)}, \\ 2Me_1^{nr} \left\{ 1 - \left(\frac{c}{2Me_1^{nr}}\right) \right\} - \frac{2}{N+1}Me_1^{nr} \left\{ 1 - \left(\frac{c}{2Me_1^{nr}}\right)^{N+1} \right\} & \text{if } S > \frac{Nc^2}{2M(N-1)}, \end{cases} \end{aligned} \right]
\end{aligned}$$

where $e_1^{nr} = \min\left\{\frac{N-1}{Nc}S, 1\right\}$.

For $S \leq \frac{Nc^2}{2M(N-1)}$, $[V^d(R) - V^d(NR)]$ is linearly increasing in S . For $\frac{Nc^2}{2M(N-1)} < S \leq \frac{Nc}{(N-1)}$, it is concave in S and for $S > \frac{Nc}{(N-1)}$ it is constant. The maxima is achieved at:

$$\frac{c}{2Me_1^{nr}} = \left\{ 1 - \frac{N+1}{N} \frac{c}{2M} \right\}^{\frac{1}{N+1}},$$

and

$$\begin{aligned}
& \max[V^d(R) - V^d(NR)] \\
&= \Delta B + 2\left(\frac{N^2}{N+1} - 1\right) \left\{ 1 - \left(\frac{c}{2}\right)^{N+1} \right\} + (2 - c) - c \left(1 - \frac{N+1}{N} \frac{c}{2M} \right)^{\frac{N}{N+1}}.
\end{aligned}$$

Also, it can be checked that $[V^d(R) - V^d(NR)]_{S=0} > [V^d(R) - V^d(NR)]_{S=\frac{Nc}{(N-1)}}$.

Further, $[V^d(R) - V^d(NR)]_{S=0}$ is increasing in ΔB and decreasing in c and $[V^d(R) - V^d(NR)]_{S=\frac{Nc}{(N-1)}}$ is decreasing in M , increasing in ΔB and decreasing in

c . Define $\Delta \hat{B}$ such that:

$$\Delta \hat{B} = \max \left\{ 0, 2(c-1) - 2\left(\frac{N^2}{N+1} - 1\right) \left\{ 1 - \left(\frac{c}{2}\right)^{N+1} \right\} \right\}.$$

To reduce the number of cases to consider, we assume that $c \leq 2$ and $\Delta B \geq \Delta \hat{B}$ so that $[V^d(R) - V^d(NR)]_{S=0} \geq 0$.

Further, define $\hat{M}_1$ so that $[V^d(R) - V^d(NR)]_{S=\frac{Nc}{(N-1)\hat{M}_1}} = 0$. Hence, for all $M \leq \hat{M}_1$, we have $V^d(R) \geq V^d(NR)$. Now consider the case where $\Delta B \geq \Delta \hat{B}$ and $M > \hat{M}_1$. Define $\hat{S}_1 \in \left(\frac{Nc^2}{2M(N-1)}, \frac{Nc}{(N-1)}\right)$ such that:

$$\left[\Delta B - c + 2 \frac{N^2}{N+1} \left\{ 1 - \left(\frac{c}{2}\right)^{N+1} \right\} + 2 \left(\frac{c}{2}\right)^{N+1} + c \hat{e}_1^{nr} \right. \\ \left. - \frac{2}{N+1} M \hat{e}_1^{nr} \left\{ N + \left(\frac{c}{2M \hat{e}_1^{nr}} \right)^{N+1} \right\} \right] = 0,$$

where $\hat{e}_1^{nr} = \frac{N-1}{N} \frac{\hat{S}_2}{c}$. Hence, under our assumptions, $V^d(NR) > V^d(R)$ if and only if $M > \hat{M}_1$ and $S > \hat{S}_1$, otherwise we have $V^d(R) \geq V^d(NR)$.

Define

$$\hat{S}_2 = \max \left\{ 0, \frac{2N^2}{N+1} \left[(M-N) + \left\{ (N-1) - \frac{1}{N} \left(1 - \frac{1}{M^N} \right) \right\} \left(\frac{c}{2} \right)^{N+1} \right] - NB^r \right\}.$$

Hence, $V^d(R) \geq V^d(P)$ if and only if $S \geq \hat{S}_2$. Note that, in addition, if $M \leq \hat{M}_1$, then $V^d(R) \geq V^d(NR)$ and hence R is chosen.

Now suppose $M > \hat{M}_1$. Define $\hat{S}_3$ such that for all $S \geq \hat{S}_3$, we have $V^d(NR) \geq V^d(P)$.²² Then P is chosen if and only if $S < \min\{\hat{S}_2, \hat{S}_3\}$. It is easy to check that there exist only two possibilities – either $\hat{S}_2 \leq \hat{S}_3 \leq \hat{S}_1$, or $\hat{S}_1 < \hat{S}_3 < \hat{S}_2$. In the latter case, R is never chosen; P is chosen if $S < \hat{S}_3$ and NR is chosen if $S \geq \hat{S}_3$. In the earlier case, P is chosen if $S < \hat{S}_2$, R is chosen if $S \in [\hat{S}_2, \hat{S}_1]$ and NR is chosen if $S > \hat{S}_1$.

For $M \leq \hat{M}_1$, suppose $\hat{S}_3 \rightarrow \infty$. Combining the analysis above, we can summarize that P is chosen if and only if $S < \min\{\hat{S}_2, \hat{S}_3\}$; NR is chosen if and only if $S \geq \max\{\hat{S}_1, \hat{S}_3\}$; in all other cases, R is chosen. ■

²² To find $\hat{S}_3$, one can find all the roots of the following equation and the highest value root gives $\hat{S}_3$:

$$\frac{x}{N} + \min \left\{ \frac{N-1}{Nc} x, 1 \right\} \left[\frac{2M}{N+1} \left\{ N + \left(\frac{c}{2M \min \left\{ \frac{N-1}{Nc} x, 1 \right\}} \right)^{N+1} \right\} - c \right] \\ = \frac{2M}{N+1} \left\{ N + \left(\frac{c}{2M} \right)^{N+1} \right\} - (B^r - \Delta B) - c.$$

If there does not exist any root of the equation above, then $\hat{S}_3 = 0$.

Proof of Proposition 5. a) It can be checked that the relationship between $V^o(P)$, $V^o(R)$ and $V^o(NR)$ with respect to S , ΔB , c and M is the same as the relationship between $V^d(P)$, $V^d(R)$ and $V^d(NR)$.

$[V^o(R) - V^o(P)] = N[V^d(R) - V^d(P)] + NE[(M - N)D - c]e_2'$, which is increasing in S as with $[V^d(R) - V^d(P)]$.

$[V^o(R) - V^o(NR)] = N[V^d(R) - V^d(NR)] + NE[(M - N)D - c]e_2'$ has the same properties with respect to S , ΔB , c and M as $[V^d(R) - V^d(NR)]$.

Finally, $[V^o(NR) - V^o(P)] = N[V^d(NR) - V^d(P)]$ which is also increasing in S .

Further, under our restriction $\Delta B \geq \hat{\Delta B}$ so that $[V^d(R) - V^d(NR)]_{S=0} \geq 0$, we have $[V^o(R) - V^o(NR)]_{S=0} \geq 0$ as well. Thus the choice between efficient licenses is qualitatively identical to the equilibrium choice between licenses. However, the cutoff levels of S and M that define the efficient license differ from the cutoffs in equilibrium.

b) Since $V^o(R) > NV^d(R)$, for every parameterization where the developers are indifferent between R and NR or P , the social planner strictly prefers R over NR or P . In other words, if $V^d(R) = V^d(NR)$, then $V^o(R) > V^o(NR)$. Similarly, if $V^d(R) = V^d(P)$, then $V^o(R) > V^o(P)$. Thus the range of S and M where the market chooses R is necessarily smaller than the range of parameters where the social planner chooses R . ■

References

- Amadeo, R. 2013. "Google's Iron Grip on Android: Controlling Open Source by Any Means Necessary." *Ars Technica*, October 20. <http://arstechnica.com/gadgets/2013/10/googles-iron-grip-on-android-controlling-open-source-by-any-means-necessary/>.
- Atal, V., and K. Shankar. 2014. "Open Source Software: Competition with a Public Good." *Atlantic Economic Journal* 42 (3): 333–45.
- Athey, S., and G. Ellison. 2014. "Dynamics of Open Source Movements." *Journal of Economics and Management Strategy* 23 (2): 294–316.
- Bar-Isaac, H., I. Jewitt, and C. Leaver. 2014. "Asymmetric Information and Adverse Selection." Working Paper. <http://www.economics.ox.ac.uk/materials/papers/13233/paper695.pdf>.
- Bergstrom, T., L. Blume, and H. Varian. 1986. "On the Private Provision of Public Goods." *Journal of Public Economics* 29 (1): 25–49.
- Blatter, M., and A. Niedermayer. 2008. "Informational Hold-Up, Disclosure Policy, and Career Concerns on the Example of Open Source Software Development." NET Institute Working Paper No. 08–06. http://www.netinst.org/Blatter_Niedermayer_08-06.pdf.
- Colazo, J., and Y. Fang. 2009. "Impact of License Choice on Open Source Software Development Activity." *Journal of the American Society for Information Science and Technology* 60 (5): 997–1011.
- Comino, S., and F. M. Manenti. 2011. "Dual Licensing in Open Source Software Markets." *Information Economics and Policy* 23 (3): 234–42.

- Comino, S., F. M. Manenti, and M. L. Parisi. 2007. "From Planning to Mature: On the Success of Open Source Projects." *Research Policy* 36 (10): 1575–86.
- D'Antoni, M., and M. A. Rossi. 2007. "Copyright vs. Copyleft Licensing and Open Source Development." Working Paper. <http://www.econ-pol.unisi.it/quaderni/510.pdf>.
- Gaudeul, A. 2005. "Public Provision of a Private Good: What Is the Point of the BSD License?" Working Paper. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=933631.
- Ghosh, S., and M. Waldman. 2010. "Standard Promotion Practices Versus Up-or-Out Contracts." *Rand Journal of Economics* 41 (2): 301–25.
- Harhoff, D., J. Henkel, and E. von Hippel. 2003. "Profiting from Voluntary Information Spillovers: How Users Benefit by Freely Revealing Their Innovations." *Research Policy* 32 (10): 1753–69.
- Holmstrom, B. 1999. "Managerial Incentive Problems: A Dynamic Perspective." *Review of Economic Studies* 66 (1): 169–82.
- Johnson, J. P. 2002. "Open Source Software: Private Provision of a Public Good." *Journal of Economics and Management Strategy* 11 (4): 637–62.
- Lerner, J., and J. Tirole. 2002. "Some Simple Economics of Open Source." *Journal of Industrial Economics* 50 (2): 197–234.
- Lerner, J., and J. Tirole. 2005. "The Scope of Open Source Licensing." *Journal of Law, Economics and Organization* 21 (1): 20–56.
- Niedermayer, A. 2013. "On Platforms, Incomplete Contracts and Open Source Software." *International Journal of Industrial Organization* 31 (6): 714–22.
- Raymond, E. S. 1999. *The Cathedral and the Bazaar*. Sebastopol, CA: O'Reilly Media Inc.
- Reinganum, J. 1989. "The Timing of Innovation: Research, Development and Diffusion." *Handbook of Industrial Organization* 1: 849–908.
- Schmidt, K. M., and M. Schnitzer. 2003. "Public Subsidies for Open Source? Some Economic Policy Issues for the Software Market." *Harvard Journal of Law and Technology* 16 (2): 473–505.
- Spence, M. 1973. "Job Market Signaling." *Quarterly Journal of Economics* 87 (3): 355–74.
- Spiegel, Y. 2009. "The Incentive to Participate in Open Source Projects: A Signaling Approach." NET Institute Working Paper No. 05–23. <http://www.tau.ac.il/126spiegel/papers/open-source16-Qtex.pdf>.
- Subramaniam, C., R. Sen, and M. L. Nelson. 2009. "Determinants of Open Source Software Project Success: A Longitudinal Study." *Decision Support Systems* 46 (2): 576–85.
- Valimaki, M. 2003. "Dual Licensing in Open Source Software Industry." *Systèmes d'Information et Management* 8 (1): 63–75.
- Zabojnik, J., and D. Bernhardt. 2001. "Corporate Tournaments, Human Capital Acquisition and the Firm Size-Wage Relation." *Review of Economic Studies* 68 (3): 693–716.