Development costs and open source software

Xiaopeng Xu*
University of California-Berkeley
xxu94@yahoo.com

Abstract

This paper analyzes the effect of the development cost on an open source software enhancement that is developed by individual programmers in an Internet community. It considers a situation in which each programmer's cost of development is common knowledge but his valuation of the enhancement is his private information, with other programmers knowing about only its distribution. Depending on the distribution functions of programmers' valuations of the enhancement, a programmer with a lower development cost may have a less incentive to develop. As the development cost decreases, the enhancement may be less likely to be developed, and some programmers may be worse off.

* I thank Eric von Hippel for encouragement and Justin Johnson and Thomas Marschak for helpful comments.

1. Introduction

The recent movement of open source software development has attracted a lot attention not only of the computing community, but also of the media, of economists and, more recently, of politicians.[1] In many countries, there are political initiatives trying to provide public support for open source, e.g., by paying direct subsidies to open source projects or by requiring government agencies and schools and universities to replace propriety software by open source software whenever possible (Schmidt and Schnitzer, 2002).

An open source process involves a large number of programmers scattered across the world sharing the source code to develop and refine pre-existing programs.[2] An experienced programmer who has access to the source code of a program is able to figure out exactly how the program works. He can then exert effort to improve or enhance the pre-existing program. Any developed software can be freely distributed through the Internet to other programmers in the Internet community and will be distributed if developed (Johnson, 2002).[3]

An open source software enhancement or refinement developed by a programmer is thus a public good. To an economist, the behavior of individual programmers engaged in open source processes is startling (Lerner and Tirole, 2002). Why should thousands of top-notch programmers contribute freely to the provision of the public good? This is an important issue to address. Lerner and Tirole make an attempt toward answering this question. They argue that career concerns can to some extent explain the behavior of individual programmers. This is echoed by some practitioners of open source software such as Eric Raymond (1999). Harhoff et al. (2000) consider other individual motivations related to the increased diffusion of freely revealed innovations. Bessen (2002) focuses on the participation of consumers, both individuals and firms, with complex needs. He argues that complexity makes a difference and that open source allows consumers to meet their needs by customizing the code themselves.

[1] The open source movement was started in 1998 by a number of prominent computer "hackers" such as Bruce Perens and Eric Raymond (von Hippel, 2002).
[2] Well-known examples of open source software are the GNU/Linux computer operating system, Perl programming language, and Internet e-mail engine SendMail (Raymond, 1999).
[3] Kollock (1999) argues that the advent of the Internet has been a boon to the development of open source development projects.

This paper sidesteps the question of group formation, but focuses on analyzing the behavior of the individual programmers who are in a *given* open source group.[4] Specifically, it addresses the following questions. Ceteris paribus, will a programmer who has a lower development cost be more likely to develop a program? Will a program be more likely to be developed when all programmers have lower development costs? Finally, will all the programmers be better off as their development costs decrease?

Conventional wisdom may lead one to answer these questions affirmatively. The paper challenges such wisdom. In a model where each programmer's valuation of the open source development or enhancement is his private information, I show that the answers to the questions depend critically on the distributions of valuations. Specifically, I obtain the following results. First, a programmer with a higher development cost may be more likely to develop. Second, the enhancement program may be less likely to be developed when all programmers have lower development costs. Finally, some of the programmers may be worse off, as their development costs decrease.

The reason for the results is simple. Developing the software enhancement is essentially an R&D activity. The provision of the enhancement, a public good, thus is of "max" or "best-shot" type. So long as there is one programmer who develops, the public good is provided; additional developers add nothing to the provision of the good. Simply put, only the best effort counts. And if every programmer can exert the best effort, then a programmer will be less likely to develop if he thinks that other programmers are more likely to develop. In other words, everyone has an incentive to be a free-rider. The free-rider problem is well known in the public economics literature. The problem is particularly severe when the public good is of best-shot type (Hirshleifer, 1983). In the development of the open source software enhancement, programmers behave strategically, each programmer's decision to develop depends on his belief about the probability that other programmers develop. The marginal benefit of development to a programmer is his valuation of the enhancement times the probability that no one else develops. Therefore, there is no simple link between the incentive to develop and the cost of development for each programmer, because a programmer develops if and only if his cost of development is no greater than the marginal benefit, rather than his valuation, of the enhancement.

[4] For example, 15 programmers did much of the initial coding of Apache. And during the first three years of Apache, 388 developers contributed 6,092 feature enhancements and fixed 695 bugs (Mockus et al., 2000).

The second result of the paper has important policy implications. Some firms pay their employees to work on open source software, because they believe that they will benefit from the enhancement. Similarly, the government and not-for-profit organizations have supported open source projects either by offering direct subsidies to the projects or by employing computer experts at universities or government agencies to support it (Schmidt and Schnitzer, 2002). The intention of such support, effecting to reduce programmers' development costs, is clearly to promote open source software developments. Unfortunately, the end result of the support may not promote but rather hinder the open source program enhancement.

Schmidt and Schnitzer (2002) discuss the implication of direct subsidies (as well as other forms of government support) for open source. They argue that the government should restrict itself to subsidizing basic research, which is a public good with strong positive external effects that will not be provided by the market. The development of most software projects, however, is applied R&D, which can be provided by the market. They also raise another problem with public subsidies to applied R&D in that it invites rent seeking activities. This leads well-intended projects to be captured by large companies managing to acquire vast amount of public subsidies as happened in the space, defense and nuclear industries. My analysis goes one step further. Even if there is no rent-seeking behavior and an open source project has the feature of basic research, public support may, due to the strategic interaction among programmers, have the unintended effect in that it may hinder rather than foster the enhancement of open source software.

The paper is closely related to Johnson (2002), who builds a model of open source software, in which individual user-programmers decide whether to invest their own time and effort to develop a software enhancement that will become a public good if so developed. Each of the programmers has a private cost of working on the project and a private value of using it; both of which are private information. The main focus of his paper is on the relationship between the probability of innovation and the number of programmers who are symmetric ex ante. I consider a modified version of his model in which each programmer has private information about his valuation of an open source program enhancement, but his development cost is common knowledge. Programmers may differ in their development costs, or their valuations of the enhancement may be differently distributed. The focus of my paper is on the effect of development costs on each individual's incentive to innovate, and on the probability that the innovation is made, as well as on the programmers' expected utility. The two papers are thus complementary.

The analysis of the paper has applications to other economic problems. For example, a large multinational corporation may have several research firms that simultaneously engage in an R&D activity to carry out a corporation-specific technological innovation. The rule of the corporation is that any innovation by one of the firms is shared fully with other firms, while the firm's R&D cost is not shared (Mas-Collel, Whinston and Green, 1995, pp. 256-257). To promote innovation, the headquarter of the corporation may subsidize the firms. Unfortunately, my analysis shows that the well-perceived policy may lead to an unintended result of hindering the prospect of innovation.

Another application is to the market for evaluations (Avery, Resnick and Zeckhauser, 1999). A group of consumers shares product evaluations which may be treated as public goods; evaluations are non-rival if the good being evaluated has elastic supply (e.g., a book or an appliance), and each individual can benefit from an evaluation without reducing its value to anyone else. Recent developments in computer networks have dramatically driven down the cost of distributing information. One may naturally think that these developments will lead more information or evaluations to be distributed. My analysis shows that the answer may not be so straightforward as long as there is still a positive cost of distribution.

The remainder of the paper is organized as follows. Section 2 sets up and analyzes the model, and Section 3 concludes.

2. The Model

There are $N > 1$ software programmers in an Internet community, each deciding simultaneously whether to expend effort and time to develop an enhancement or refinement of a pre-existing software application, the source code of which is open. In other words, each programmer makes a binary choice: $s_i \in \{0, 1\}$, where 1 means "develop" and 0 means "don't develop".

The enhancement, once developed, will then become a public good, available to all programmers in the community. Developing the software enhancement is essentially an R&D activity. The provision of the enhancement, the public good, thus is of "max" or "best-shot" type.

So long as there is one programmer choosing to develop, the enhancement is made; additional developers add further value to the innovation.

Programmers derive benefits from the enhancement; they may be driven to write software for their own use, such as facilitating their own work or debugging (Johnson, 2002; von Hippel, 2002). The benefit of the enhancement to programmer i is b_i, which is his private information. Other players know about only the distribution of b_i. The cumulative distribution function for b_i is $F_i(b_i)$, and the corresponding density function is $f_i(b_i) > 0$, for $b_i \in [\underline{b}_i, \bar{b}_i]$, $0 \leq \underline{b}_i < \bar{b}_i$. Assume that programmers' benefits are independently distributed. A programmer working on a software development project incurs an opportunity cost of his time; he cannot work on other projects while he is working on this project. The development cost for programmer i is $c_i > 0$, which is common knowledge. This specification differs from that of Johnson (2002) in which each programmer's valuation of the enhancement and cost of development are his private information.

Given the best-shot feature of the enhancement, programmer i's utility function is

$$U_i = \max\{s_i, \ldots, s_N\} b_i - s_i c_i. \tag{1}$$

If at least one of the N programmers develops, programmer i receives his privately known benefit, b_i, but he incurs his development cost, c_i, if and only if he is a developer.

Assumption 1. $\underline{b}_i < c_i < \bar{b}_i$, i = 1, …, N.

This assumption implies that, if programmer i were to exist in isolation, he would develop if and only if his valuation of the enhancement is no less than his cost of development, $b_i \geq c_i$, and his ex ante probability to develop would be $1 - F_i(c_i)$, which is strictly between 0 and 1. When he interacts with other programmers, programmer i's decision to develop depends on his belief about the probabilities that other programmers develop. Let p_j be programmer i's belief about the ex ante probability that programmer j develops, $j \in \{1, \ldots, N\}\backslash\{i\}$. If programmer i develops, his utility or payoff is $b_i - c_i$. If programmer i does not develop, his expected payoff is $[1 - \prod_{j=1, j\neq i}^{N}(1 - p_j)] b_i$, which is equal to the product of his valuation of the enhancement and the probability that at least one of the other N - 1 programmers develops.

Clearly, programmer i develops, i.e., $s_i = 1$, if and only if $\prod_{j=1, j\neq i}^{N}(1-p_j) b_i \geq c_i$. In other words, programmer i's development decision obeys a cutoff rule: he develops if and only if his valuation of the enhancement is no less than a cutoff level, $b_i^* = c_i / \prod_{j=1, j\neq i}^{N}(1-p_j)$. This is true for all programmers. Thus, the ex ante probability that programmer i develops is $p_i = 1 - F_i(b_i^*)$, i = 1, …, N.

Note that $b_i^* > c_i$. In the presence of other programmers, each programmer is less likely to develop than he would be in isolation, in hoping that others will develop so that he can be a free-rider. In equilibrium, we have, for i = 1, …, N,

$$b_i^* \prod_{j=1, j\neq i}^{N} F_j(b_j^*) = c_i. \qquad (2)$$

Equilibrium conditions (2) are general, allowing different development costs and distributions of valuations of the enhancement among all the programmers. With restrictions on the distributions of valuations or development costs, we can use (2) to address many important issues. For instance, one may intuitively think that, ceteris paribus, a programmer with a lower development cost may be more likely to develop. Unfortunately, this intuition is generally incorrect, as I will show below.

Assume that all programmers' benefits of the enhancement are identically and independently distributed. Denote the common distribution function F(b), $b \in [\underline{b}, \bar{b}]$, $\bar{b} > \underline{b} \geq 0$. From (2), one can easily see that, for any two programmers i and j, $i \neq j$, we have

$$c_i F(b_i^*)/b_i^* = c_j F(b_j^*)/b_j^*. \qquad (3)$$

Let $G(b) = F(b)/b$. Clearly, if $G'(b) = [bf(b) - F(b)]/b^2 < 0$, then $b_i^* > b_j^*$, when $c_i > c_j$, implying that a programmer with a lower development cost is more likely to develop. On the other hand, if $G'(b) > 0$, then $b_i^* < b_j^*$, when $c_i > c_j$, implying that a programmer with a lower development cost is less likely to develop.

The reason for the result is as follows. Each programmer i's marginal benefit of development is his valuation of the enhancement times the probability that no one else develops. His marginal cost of development is simply his cost of development. In equilibrium, a programmer develops if and only if his marginal benefit of development is no less than his cost of development. Consequently, he develops if and only if his valuation of the enhancement is no less than a cutoff level. For two programmers with different development costs, there are two possibilities. In the first which complies more with intuition, the programmer with a lower (higher) development cost has a lower (higher) cutoff level, indicating that he is more (less) likely to develop. In the second which is in contrast with intuition, the programmer with a lower (higher) development cost has a higher (lower) cutoff level, indicating that he is less (more) likely to develop. Depending on the distribution function of development costs, either possibility can take place.

It is easy to identify distribution functions such that G'(b) is either positively or negatively signed. For example, let $F(b) = b^k$, $b \in [0, 1]$, $k \neq 1$. Then, $G'(b) > 0$, when $k > 1$, but $G'(b) < 0$, when $k < 1$. Indeed, one can easily find commonly used distribution functions F(b) such that G'(b) can be either positive or negative. For example, if F(b) is uniformly distributed, for $b \in [\underline{b}, \bar{b}]$, $\bar{b} > \underline{b} > 0$, then $G'(b) > 0$, implying that a programmer with a lower development cost is less likely to develop. If F(b) is exponentially distributed over $[0, \infty)$, then $G'(b) < 0$, implying that a programmer with a lower development cost is more likely to develop.

Let $H(b) = F(b)/[bf(b)]$. Because $F(b)/f(b)$ is the hazard rate, I call H(b) the unit hazard rate. Proposition 1 summarizes the above analysis.

Proposition 1. Suppose that programmers' valuations of the enhancement are independently and identically distributed. If the unit hazard rate for the common distribution is greater (smaller) than 1, then a programmer with a lower development cost is more (less) likely to develop.

A successful open source project requires a credible leader, and most leaders of open source projects are the programmers who developed the initial code for the projects (Lerner and Tirole, 2002). The development cost for a leader may be higher than that for other programmers, his followers. Being leaders, they may have better outside opportunities and hence higher opportunity costs of working on open source projects. Proposition 1 provides an explanation for the leader's behavior. Even if his valuation of the open source software development is identically distributed with other programmers, the leader may be more likely to develop when the unit hazard rate of the distribution is less than 1, even though he has a higher development cost.

Aiming to foster the open source movement, the government and not-for-profit organizations in many countries make subsidies to open source projects, so do some private firms (Lerner and Tirole, 2002; Schmidt and Schnitzer, 2002). Such policy effects to reduce development costs. The policy is clearly based on the belief that reductions in development costs are conducive to open source software development. But the important question is: Does the belief make economic sense? I show that the answer to this question depends critically on heterogeneity in programmers' valuation distributions.

To simplify the analysis, assume that all programmers have the same development cost, i.e., $c_i = c > 0$, $i = 1, \ldots, N$. It is clear from (2) that b_i^* is a function of c, $b_i^* = b^*(c)$, for all i. The ex ante probability that the enhancement is not developed is $Q(c) = \prod_{i=1}^{N} F_i(b_i^*(c))$. In the following, I sometimes choose to suppress the argument of $F_i(b_i^*(c))$ and $f_i(b_i^*(c))$, as there should be no confusion, given the context. Differentiating Q with respect to c, we have

$$Q'(c)/Q(c) = \sum_{i=1}^{N} \frac{f_i}{F_i} db_i^*/dc. \quad (4)$$

Clearly, if $db_i^*/dc > 0$ for all i, $Q'(c) > 0$, implying that a lower (common) development cost will result into a higher probability that the enhancement is developed.

If $F_i(.) = F_i(.) = F(.)$, for all i and j. Then, in symmetric equilibrium, $b_i^* = b_j^* = b^*$ (this is guaranteed if G(b) is monotonic), and it follows from (2) that b^* satisfies that $b^*F(b^*)^{N-1} = c$. Obviously, $db^*/dc > 0$. Hence, $dQ/dc > 0$.

Proposition 2. Suppose that programmers are ex ante symmetric in that they have the same development cost and their valuations of the enhancement are independently and identically distributed. Then, in symmetric equilibrium, as the development cost decreases, the enhancement is more likely to be developed.

More generally, we have $F_i(b_i^*(c))/b_i^*(c) = F_j(b_j^*(c))/b_j^*(c)$. Differentiating both sides of this equation with respect to c, one can readily check that $[f_i(b_i^*(c))b_i^*(c) - F_i(b_i^*(c))]/b_i^*(c)^2 \; db_i^*/dc = [f_j(b_j^*(c))b_j^*(c) - F_j(b_j^*(c))]/b_j^*(c)^2 \; db_j^*/dc$. Thus, if $H_i(b_i) - 1 = F_i(b_i)/[b_i f_i(b_i)] - 1$ is uniformly and identically signed, for all i, then db_i^*/dc is uniformly and identically signed. Using this fact together with (2), one can easily show that $db_i^*/dc > 0$, for all i. Thus, we have

Proposition 3. If $H_i(b_i) - 1$ is uniformly and identically signed, for all i, then $Q'(c) > 0$.

However, when $H_i(b_i) - 1$ and $H_j(b_j) - 1$ are uniformly but not identically signed, then db_i^*/dc and db_j^*/dc are oppositely signed and one of them must be negative, implying that one of the them is less likely to develop, as the development cost decreases. Indeed, a lower development cost may result into a lower probability that the enhancement is developed! To bring out the result most straightforwardly, I consider the case of two programmers, $N = 2$, whose benefits are distributed according to cumulative functions $F_1(b_1)$ and $F_2(b_2)$, respectively.

It follows immediately from (2) that the cutoff levels, b_1^* and b_2^*, for the two programmers satisfy that $b_1^*F_2(b_2^*) = c$ and $b_2^*F_1(b_1^*) = c$. Clearly, b_1^* and b_2^* are functions of c. Totally differentiating both sides of the two equations with respect to c, we have

$$\begin{pmatrix} F_2 & b_1^* f_2 \\ b_2^* f_1 & F_1 \end{pmatrix}\begin{pmatrix} db_1^* \\ db_2^* \end{pmatrix} = \begin{pmatrix} dc \\ dc \end{pmatrix}. \tag{5}$$

If $F_1F_2 - b_1^*b_2^*f_1f_2 \neq 0$, then one can solve (5) and obtain $db_i^*/dc = (F_i - b_i^*f_j)/(F_1F_2 - b_1^*b_2^*f_1f_2)$, $i = 1, 2$, $j = 3 - i$.

The ex ante probability that the enhancement is not developed is $Q(c) = F_1(b_1^*(c))$ $F_2(b_2^*(c))$. Differentiating Q with respect to c and plugging into db_1^*/dc and db_2^*/dc, we have

$$\begin{aligned} Q'(c) &= F_2f_1db_1^*/dc + F_1f_2db_2^*/dc \\ &= (F_1F_2f_1 - b_1^*f_1f_2F_2 + F_1F_2f_2 - b_2^*f_1f_2F_1)/(F_1F_2 - b_1^*b_2^*F_1F_2). \\ &= [(H_1 - 1)F_2/b_2^* + (H_2 - 1)F_1/b_1^*]/(H_1H_2 - 1), \end{aligned} \tag{6}$$

where $H_i = F_i(b_i^*(c))/[b_i^*(c)f_i(b_i^*(c))]$. Note that, in equilibrium, $F_1/b_1^* = F_2/b_2^*$. Hence, we have the following result.

Proposition 4. $\text{Sign}(dQ/dc) = \text{Sign}((H_1 + H_2 - 2)/(H_1H_2 - 1))$.

From Proposition 4, we know immediately that, if $H_1 = H_2$, then $dQ/dc > 0$, implying that a lower development cost will result into a higher probability that the enhancement is developed. This is, of course, not surprising at all, given Proposition 2. On the other hand, when $H_1H_2 < 1$ and $H_1 + H_2 > 2$, $dQ/dc < 0$. A necessary condition for this to happen is that $H_i > 1$ and $H_j < 1$, $i = 1, 2$, $j = 3 - i$. In this case, a lower (common) development cost leads to a lower probability that the enhancement is developed.

It is important to point out that Proposition 4 is valid only for interior equilibrium (which we have implicitly assumed in the analysis), in which both programmers develop with positive probability. If one of the programmers never develops, then, it is clear that the other programmer is more likely to develop as the development cost decreases, leading to a higher probability that the enhancement is developed. An example of this is as follows. Let $F_1(b_1) = b_1^{1/j}$, $0 < j < 1$, $b_1 \in [0, 1]$, $F_2(b_2) = b_2^{1/k}$, $k > 1$, $b_2 \in [0, 1]$, and $0 < c < 1$. Simple calculation shows that $H_1(b_1) \equiv j$ and $H_2(b_2) \equiv k$. Let $jk < 1$ and $j + k > 2$. So, the conditions in Proposition 4 are satisfied. One can easily verify that, for $b_2 \geq c^{1-1/j}$, there is a unique equilibrium characterized by $b_1^* = 1$ and $b_2^* = c$.[5] In equilibrium, $Q(c) = c^{1/k}$. Clearly, $Q'(c) > 0$.

I now give an example in which a smaller (common) development cost leads to a lower probability that the enhancement is developed.

Example 1. Let $F_1(b_1) = b_1^{1/2}/2$, $b_1 \in [0, 4]$, and $F_2(b_2) = b_2 - 1$, $b_2 \in [1, 2]$, and $1 < c < 2$. It is easy to verify that $H_1(b_1) \equiv 2$, and $H_2(b_2) = (b_2 - 1)/b_2$. For $1 < b_2 < 2$, $0 < H_2(b_2) < 1/2$. Hence, we have $H_1H_2 < 1$ and $H_1 + H_2 > 2$. Thus, in interior equilibrium, a lower development cost leads to a lower probability that the enhancement is developed.

All we need to check is whether there exists interior equilibrium for some values of the development cost. If follows from (2) that $b_1^*(b_2^* - 1) = c$ and $b_2^* b_1^{*1/2}/2 = c$. Solving b_1^* and b_2^*, we have $b_1^*(c) = c/[2c - 1 - 2\sqrt{c(c-1)}]$ and $b_2^*(c) = 2c - 2\sqrt{c(c-1)}$. In interior equilibrium, $c < b_1^*(c) < 4$ and $c < b_1^*(c) < 2$. One can easily verify that, when $1 < c < 4/3$, the above two inequalities are satisfied. Further, one can show that, for $c > 1$, $db_1^*/dc > 0$ and $db_2^*/dc < 0$.

It follows from Proposition 4 immediately that, for $1 < c < 4/3$, a reduction in the development cost results into a lower probability that the enhancement is made.

We thus see that, depending on the distributions of programmers' valuations of the enhancement, the innovation may be either more or less likely to be made available. This is the main result of the paper. This result has important policy implications. Private firms, the government, and not-for-profit organizations, aiming to promote open source development

[5] When $b_2 < c^{1-1/j}$, besides the above equilibrium, there is another equilibrium in which $b_1^* = c$ and $b_2^* = 1$.

processes, have made subsidies to open source projects. My analysis indicates that such policy may not serve the intended purpose, but rather hinder the processes.

Finally, let us study the effect of the development cost on programmers' payoffs. Programmer i's expected payoff is

$$EU_i = p_i EU_i\Big|_{b_i \geq b_i^*} + (1 - p_i)[1 - \prod_{j=1, j\neq i}^{N}(1-p_j)]EU_i\Big|_{b_i < b_i^*}$$

$$= \int_{b_i^*}^{\bar{b}_i}(b_i - c)\, dF_i(b_i) + [1 - \prod_{j=1, j\neq i}^{N} F_j(b_j^*)] \int_{\underline{b}_i}^{b_i^*} b_i\, dF_i(b_i). \qquad (7)$$

Differentiating EU_i with respect to c, we have

$$d(EU_i)/dc = \partial(EU_i)/\partial c + \partial(EU_i)/\partial b_i^*\, db_i^*/dc + \sum_{j=1, j\neq i}^{N} \frac{\partial(EU_i)}{\partial b_j^*} db_j^*/dc. \qquad (8)$$

It is easy to check that $\partial(EU_i)/\partial b_i^* = 0$, as programmer i has optimally chosen b_i^* to maximize his payoff. The first term is the direct effect of the development cost on programmer i's payoff, while the third term is the indirect, strategic effect of the development cost on programmer i's payoff. Simple calculation shows that $\partial(EU_i)/\partial c = F_i(b_i^*(c)) - 1 < 0$. The direct effect of the development cost on programmer i's payoff is always beneficial, as a lower development cost leads to a higher payoff for programmer i, given other programmers' development choices.

Differentiating EU_i with respect to b_j^*, we have $\partial(EU_i)/\partial b_j^* = - f_j(b_j^*) \prod_{k=1, k\neq i,j}^{N} F_k(b_k^*) \int_{\underline{b}_i}^{b_i^*} b_i\, dF_i(b_i) < 0$. If programmer j is more likely to develop, i.e., b_j^* decreases, programmer i's expected payoff increases. However, we have shown that, depending on the distribution functions of programmers' valuations, db_j^*/dc can be either positively or negatively signed. When $db_j^*/dc < (>) 0$, programmer j's change of behavior, in response to changes in the development cost, imposes a negative (positive) external effect, $- f_j(b_j^*) \prod_{k=1, k\neq i,j}^{N} F_k(b_k^*) \int_{\underline{b}_i}^{b_i^*} b_i\, dF_i(b_i)\, db_j^*/dc$, on programmer i's payoff. The indirect effect is the sum of all the external effects imposed on programmer i by the other N - 1 programmers.

From Propositions 2 and 3, we know that, if programmers are ex ante symmetric, or if $H_i(b_i) - 1$ is uniformly and identically signed for all i, then $db_i^*/dc > 0$, i = 1, …, N. Thus, each programmer is better off, as the (common) development cost decreases. I summarize the result by Proposition 5.

Proposition 5. When programmers are symmetric ex ante, or when the difference between the unit hazard rate of valuation and 1 is uniformly and identically signed for all programmers, a lower common development cost leads to a higher expected payoff for all programmers.

When $H_i(b_i)$ - 1 is not identically signed for all programmers, then the indirect effect of reductions in the development cost is detrimental for some programmer, say, programmer i. Hence, $d(EU_i)/dc$ can be either negatively or positively signed, depending on whether the direct effect dominates or is dominated by the indirect effect. The following example illustrates this.

Example 1 (continued). Recall that $db_2^*/dc < 0$. Hence, $\partial(EU_1)/\partial b_2^*\ db_2^*/dc > 0$. When $\partial(EU_1)/\partial b_2^*\ db_2^*/dc > -\ \partial(EU_1)/\partial c$, $d(EU_1)/dc > 0$, indicating that programmer 1 is worse off ex ante, as the development cost decreases.

Simple calculation shows that $\partial(EU_1)/\partial c = F_1(b_1^*(c)) - 1 = c/[2c - 2\sqrt{c(c-1)}\,] - 1$, and $\partial(EU_1)/\partial b_2^*\ db_2^*/dc = -\ f_2(b_2^*(c)) \int_{\underline{b}_1}^{b_1^*} b_1 dF_1(b_1)\, db_2^*/dc = -\ [2 - (2c - 1)/\sqrt{c(c-1)}\,]\{c/[2c - 1 - 2\sqrt{c(c-1)}\,]\}^{3/2}/6$. With some algebra, one can show that, for $c > 1$, $\partial(EU_1)/\partial b_2^*\ db_2^*/dc > -\ \partial(EU_1)/\partial c$. Hence, $d(EU_1)/dc > 0$, for $1 < c < 4/3$, implying that, in interior equilibrium, a reduction in the development cost makes programmer 1 strictly worse off.

3. Conclusion

This paper attempts to answer the following questions in open source software development. Ceteris paribus, will a programmer with a lower development cost more likely to develop? Whether more of the enhancement will be developed as programmers' development costs decrease? Will all programmers be better off as their development costs decrease? Conventional wisdom may lead one to answer these questions affirmatively. However, I show in the paper that the answers to the questions are not so straightforward.

There are two salient features of an open source software development or enhancement. First, it is a public good. The enhancement benefits all programmers in a group in a non-rivalrous way. Second, it is a public good of "max" or "best-shot" type. The enhancement can be essentially developed by one programmer; additional developers, while incurring development costs, add no further value to the enhancement. The best-shot public good feature of the open source software enhancement leads to the well-known free-rider problem; each programmer wants to be a free rider, hoping that someone else will develop. The free-rider motive is the underlying force driving the results of the paper.

The main result of the paper is that, as programmers' development costs decrease, the open source enhancement may be less likely to be developed. It has important policy implications, as some private firms and government agencies as well as not-for-profit organizations, aiming to promote open source software development, make subsidies to open source projects. Such subsidies effect to reduce programmers' development costs. The analysis of paper, however, indicates that the subsidies may hinder rather than foster, as intended, the innovation of open source software programs.

The reason for the result is that programmers involved in the open source development act strategically. If a programmer perceives that other programmers are more likely to develop owing to lower development costs, he is less likely to develop. The end result may be that the program is less likely to be developed even when all programmers' development costs decrease.

References

Avery, C., Resnick, P., Zeckhauser, R., 1999. The market for evaluations. American Economic Review 89, 564-584.

Bessen, J., 2002. Open source software: free provision of complex public goods. Mimeo, Research on Innovation.

Harhoff, D., Henkel, J., von Hippel, E., 2000. Profiting from voluntary information spillovers: how users benefit by freely revealing their innovations? Mimeo, MIT.

Hirshleifer, J. 1983. From weakest-link to best-shot: the voluntary provision of public goods. Public Choice 41, 371-386.

Johnson, J., 2002. Open source software: private provision of a public good. Journal of Economics and Management Strategy, forthcoming.

Kollock, P., 1999. The economies of online cooperation: gifts and public goods in Cyberspace. In Smith, M., Kollock, P., (eds.). *Communities in Cyberspace*. London, Routledge.

Lerner, J., Tirole, J., 2002. The simple economics of open source. Journal of Industrial Economics 50, 197-234.

Mas-Collel, A., Whinston, M., Green, J., 1995. *Microeconomic Theory*. Oxford University Press, New York.

Mockus, A., Fielding, R., Herbsleb, J., 2000. A case study of open source software development: the Apache server. Available at http://opensource.mit/edu/papers/mockusapache.pdf.

Raymond, E., 1999. *The Cathedral & the Bazaar: Musing on Linux and Open Source by an Accidental Revolutionary.* Cambridge, O'Reilly.

Schmidt, K., Schnitzer, M., 2002. Public subsidies for open source? some economic policy issues of the software market. Mimeo, University of Munich.

von Hippel, E., 2002. Open source software projects as user innovation networks. Mimeo, MIT.