Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

Published online ahead of print June 16, 2011

Information Systems Research
Articles in Advance, pp. 1–10
ISSN 1047-7047 | EISSN 1526-5536

DOI 10.1287/isre.1110.0358
© 2011 INFORMS

Research Note

Lock-In Strategy in Software Competition: Open-Source Software vs. Proprietary Software

Kevin Xiaoguo Zhu, Zach Zhizhong Zhou
The Rady School of Management, University of California, San Diego, La Jolla, California 92093
{kxzhu@ucsd.edu, zzhou@rady.ucsd.edu}

Open-source software poses a serious challenge to proprietary software vendors. "Lock in customers" seems a tempting strategy for proprietary software vendors, who attempt to lock in customers by creating switching costs. This paper examines whether such a lock-in strategy will indeed benefit proprietary software vendors facing competition from open-source software, who can credibly commit future prices. Developing a two-period duopoly model in which software products are differentiated and customers are heterogeneous, we find that the lock-in strategy is actually counterproductive in competing against open-source software. In fact, giving customers the freedom of choice may end up benefiting the proprietary software vendor. In terms of the broader effect, we find that lock-in reduces overall social welfare, but certain customers may actually be better off with it. Finally, we show that the lock-in strategy works differently for different types of customers in the software market (i.e., foresighted versus myopic customers). This suggests that customer behavior could significantly alter the equilibrium strategy of software vendors.

Key words: software; competition; lock-in; open-source software; proprietary software; game theory; switching cost
History: Vallabh Sambamurthy, Senior Editor; Gautam Ray, Associate Editor. This paper was received on December 6, 2007, and was with the authors 18 months for 6 revisions. Published online in *Articles in Advance*.

1. Introduction

With the emergence of new technologies such as on-demand computing and cloud computing, "vendor lock-in" becomes one of the major concerns of chief information officers (Computer World 2004). The term "lock-in" refers to a situation in which a customer is dependent on a vendor for products and services such that he or she cannot switch to another vendor without suffering substantial costs. It is often observed in the computer industry that the substantial costs to switch between different software systems force a customer to continue to use products and services from a particular vendor (Zhu et al. 2006).

A software vendor can lock in customers in several ways: (1) by designing a system incompatible with software developed by other vendors; (2) by using proprietary standards or closed architectures that lack interoperability with other applications; (3) by licensing the software under exclusive conditions (Kucharik 2003). Then, "lock-in" can be a deliberate strategy for a software firm. Conventional wisdom suggested that such a strategy benefits software vendors because vendor lock-in may reduce the bargaining power of customers and increase that of vendors in the postadoption period; proprietary vendors may gain competitive advantages (or even monopoly power) from a lock-in strategy.

It seems intuitive that proprietary software vendors should always prefer lock-in. However, prior literature has shown that this is not always the case when the competition is between two proprietary vendors. The key reason for this is that the two vendors could engage in fierce competition for market share (e.g., through deep discounts) in anticipation of postadoption monopolistic profits. Under certain conditions, such intensified competition for market share could backfire and hurt them.

Klemperer (1987b) considers a two-period duopolistic competition with switching costs. He shows that switching costs may lead to monopoly rents, but these rents can be competed away when firms entice customers to sign up. Specifically, each firm would like to give up its first-period profit for its second-period gains *at the margin*.[1] However, the *total* first-period profit given up can be in any relationship

[1] Suppose that the decision variable of a firm is price p. The firm's total profit is $\pi = \pi_1 + \pi_2$, where π_i is the profit earned in period i ($i = 1, 2$). Then the first-order condition $d\pi/dp = d\pi_1/dp + d\pi_2/dp = 0$ leads to $-d\pi_1/dp = d\pi_2/dp$. That is, the marginal gain in the second period equals the marginal loss in the first period.

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

to the *total* second-period gains. Thus, switching costs can either help or hurt firms overall. Klemperer (1987a) uses a two-period horizontal differentiation model to study the competitiveness of markets with switching costs. A proportion of consumers' tastes for underlying product characteristics in the second period are independent of their tastes in the first period. He shows that if the proportion of such kind of consumers increases, switching costs make firms worse off. Caminal and Matutes (1990) examine two policies that create endogenous switching costs. The first policy precommits a second-period price to loyal consumers while charging a higher price to new consumers. The second policy gives coupons to loyal consumers while charging the same price to all consumers in the second period. The second policy (offering coupons) tends to generate higher equilibrium profits for firms than the first policy (price commitment). Viswanathan (2005) uses a spatial differentiation model to study channel competition and shows how switching costs affect the prices and profits of a given channel. These studies show that switching costs or lock-in may or may not benefit competing firms, depending on specificity of the competition.

As a response to the concern of "lock-in," open-source movement is gaining momentum in the software industry. Open source refers to free access, free distribution, and free modification of software source codes (OSI 2011b). According to a survey by Computer Economics (2005), the key advantage of open source is not cost savings but reduced dependence on proprietary software vendors. The main reason is that there is no forced upgrade, and the software can be supported by an open community, as opposed to a proprietary vendor. The competition in the open community reduces the price of supporting and maintaining the legacy software. In contrast, proprietary software vendors often release new versions of software, force customers to upgrade and make them more dependent on vendor support, or else the customer would be stranded with an outdated system. The Open Source Initiative, a nonprofit organization formed to promote open source, claims that "the promise of open source is . . . an end to predatory vendor lock-in" (OSI 2011a).

Earlier studies in the literature mainly focused on the competition between two proprietary products. Little is known about *the competition between not-for-profit open-source and proprietary software with lock-in*. The competition is different when an open-source software (OSS) is involved. Compared with proprietary software, OSS is developed by open-source communities, which typically are not profit-maximizing entities. Further, OSS can be freely distributed among open-source communities, whereas the price of proprietary software is set by a strategic profit-seeking vendor. The free distribution of OSS ensures that customers are able to get the software free or at a low price that covers the distribution cost. This is because it is difficult for a software vendor to get a decent profit from distributing OSS, given the fact that there are many potential competitors (that is, OSS users), who may freely distrubte OSS as well.[2]

As illustrated above, the literature on the competition between two proprietary firms (e.g., Klemperer 1987a, b) shows that the effect of lock-in is driven by two forces: loss caused by intensified competition for market share in period 1 and gains from exploiting customers in period 2. The overall effect of switching costs can benefit or hurt competing firms. It is not clear if the same logic can be applied to competition between a *strategic* proprietary software provider and a *nonstrategic* OSS. Unlike proprietary software, the OSS does not behave strategically and thus would not use deep discounts to fight for market share in the first period and then raise its price in the second period. Thus the game will be different from that analyzed in the literature. Will a lock-in strategy benefit the proprietary software vendor when it faces a competitive threat from OSS?

By developing a two-period model where target customers are differentiated by their tastes and reservation value, we show that the lock-in strategy can be counterproductive for the proprietary software vendor. A key feature of our model is that the OSS has *a nonstrategic but credible commitment* to its future prices (i.e., it is still free in the future) as guaranteed by OSS licenses such as the GNU General Public License (GPL) agreement (GNU 2007). The GPL "free distribution" term ensures that nobody gains a monopoly power as an OSS distributor. This allows users to obtain OSS at a minimum cost. The other GPL term, "free modification," ensures that nobody can offer a free software product now and then charge a high price for an upgraded version in the future. With free modification, OSS users can always write codes to upgrade the software. Nobody gains a monopoly power as a software upgrade provider. Thus, this term introduces full competition into software upgrade and ensures that the price of service in the future cannot be a monopoly price.

[2] For example, users may download the following OSS products for free from the Internet: Python (programming/scripting language), OpenOffice (office suite), Apache (Web server), LaTeX (typesetting language), Zope (Web application server), UPortal (Web portal framework), Blender (3D graphics and animation package), Mozilla (Web browser and email client), Plone (content management system), OSS version of MySQL (database), and Samba (file and print server).

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

The price commitment of OSS makes it difficult for the proprietary software vendor to capture forward-looking customers who can sense the coming danger of being locked in. If the proprietary software vendor uses the lock-in strategy, a forward-looking customer will expect to be charged a high price in the future. Then he would demand a deep discount or even a subsidy before adopting the proprietary software in the first place. We show that the total subsidies used to capture customers in the first period is so large that the lock-in strategy hurts the proprietary software vendor.

We further show that the lock-in strategy works differently for different types of customers in the software market (i.e., foresighted versus myopic customers). When customers are myopic instead of foresighted, the lock-in strategy turns out to benefit the proprietary software vendor in the competition between proprietary software and OSS—a different result than when customers are foresighted. We also examine the broader effects of lock-in on customer welfare and social welfare.

1.1. Related Literature

There is a growing literature on OSS (see von Krogh and von Hippel 2006 for a review). These studies focus on different aspects of the open-source phenomenon, especially along the lines of user-developer participation incentives. For example, Lerner and Tirole (2002) analyze OSS as private provision of a public good and suggest that participation in OSS may be driven by career benefits such as signaling skill and peer recognition. Roberts et al. (2006), through a longitudinal study of the Apache projects, examine the motivation for open-source contributors, such as status, use-value, and intrinsic and extrinsic motives. Mustonen (2002) studies open-source quality and shows that the occupational choices of programmers based on reputation incentives determine the quality of programs. Economides and Katsamakas (2006) examine platform competition where each platform can support complementary applications. Casadesus-Masanell and Ghemawat (2006) analyze the competition between Linux and Windows, which features demand-side learning and the survival of the for-profit firm is based on the existence of network effects on the demand side. "This competitive strand of research on OSS is much less developed than the organizational strand discussed above, even though the rhetoric about it . . . can get quite heated" (Casadesus-Masanell and Ghemawat 2006, p. 1074). Bitzer (2004) argues that product differentiation is critical for the proprietary software to compete with the OSS.

Our study is along this line, but it differs from prior OSS literature in the following sense: we focus on *lock-in* as a possible competitive weapon and look at this from the standpoint of a proprietary software vendor as opposed to OSS. These two clearly have different objective functions. Furthermore, we model price commitment as an important feature of the OSS, whereas the existing literature studied OSS from other aspects (e.g., incentives of contributors or demand-sided learning), as described above. Price commitment has not been formally modeled as an important feature of the OSS, but we believe that it needs to be examined carefully. Along this line, our paper fills a gap and adds new insights to the OSS literature.

The remainder of the paper is organized as follows. Section 2 describes the model setup and assumptions. Section 3 analyzes the competition between proprietary and OSS. Section 4 shows how customer behaviors influence software vendors' desire to lock-in customers. Section 5 shows the effects of lock-in on consumer surplus and social welfare. Section 6 concludes the paper. Detailed mathematical proofs are presented in the online e-companion on the journal's website.

2. The Model

Two software vendors compete in the same market. They license their software in two periods, namely, the current and future periods. Prices are announced simultaneously at the beginning of each period. Neither vendor can set prices based on customer types, because an individual customer's type is private information. Software is labeled by A or O, where A stands for proprietary software and O for open-source software. It is well known that software products tend to have low or even zero marginal costs (Shapiro and Varian 1999). For technical simplicity, marginal costs of both vendors are normalized to zero in our model. In particular, OSS can be obtained at a minimum cost (e.g., by freely downloading from host servers or purchasing a CD for a few dollars). We follow prior literature (Economides and Katsamakas 2006, Casadesus-Masanell and Ghemawat 2006) by assuming that:

Assumption 1. *The cost of obtaining OSS is normalized to zero.*

Denote p_{it} ($i = A, O$; $t = 1, 2$) as the cost of obtaining software i in period t. We have $p_{O1} = p_{O2} = 0$ by Assumption 1. Because p_{O1} and p_{O2} are not strategic decision variables in this paper, we rescale them to zero for technical convenience. Later it will be clear that even though we set p_{O1} and p_{O2} as positive constants, it will not change the major results of this paper.

OSS and proprietary software are clearly differentiated. Schmidt and Schnitzer (2002) modeled this

Copyright: INFORMS holds copyright to this Articles in Advance version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

as a horizontal differentiation. Following the literature (Caminal and Matutes 1990, Klemperer 1995, Marinoso 2001), we capture software differentiation via a *horizontal differentiation model.*

Assumption 2. *OSS and proprietary software are horizontally differentiated.*

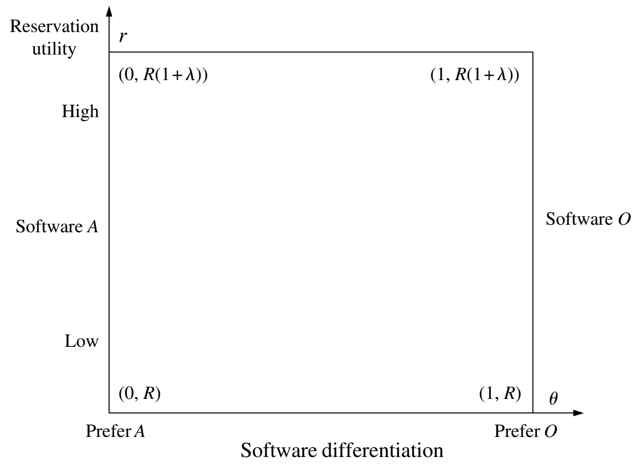
The defining characteristic of horizontal differentiation is "if all the variants of a product are sold at the same price, there is a positive demand for each of them" (Cremer and Thisse 1991, p. 383). That is, if both A and O are sold at the same price, some customers would choose A and others would choose O because they rank product features differently. For example, OSS is easier to customize and reuse, can be developed collaboratively, and helps users learn programming techniques from open-source communities. Compared with OSS, proprietary software does not open access to its source codes, thus it is difficult to be customized. Meanwhile, it tends to have a higher degree of usability in the sense of documentation and user interface and is easier to maintain by a clear vendor. Those customers who prefer a higher usability would prefer proprietary software, whereas those who want to customize software would prefer OSS, even if they are sold at the same price (Lerner and Tirole 2002). Take the choice between J2EE and .NET as an example. Estes and Maxime (2003) summarize a list of comparisons between J2EE and .NET, and then propose that the choice depends on the needs and preference of the user company. Most articles conclude that there is no clear winner or obvious choice, which conforms to the basic idea of horizontal differentiation.

Motivated by such observation in practice, we introduce another dimension of heterogeneity: *reservation utility*. This makes our model two-dimensional differentiation. It is different from the literature, which either considers one-dimensional horizontal differentiation or one-dimensional vertical differentiation. For any given software, customers may have heterogeneous use for the software. Some customers can be heavy users of the software, and others are light users. It leads to different reservation utility among customers because of their differences in valuation, budget, or cost (Gartner Research Note 2003).

Assumption 3. *Customers are vertically differentiated in their reservation utility.*

To sum up, our model captures heterogeneity along two dimensions: (1) customers' reservation utility (r) and (2) their preferences for horizontally differentiated software (θ). Mathematically, any customer can be represented by parameters (r, θ) in a two-dimentional map as illustrated in Figure 1.

Figure 1 The Customer Model



Along the horizontal axis, A is located at 0 and the competing software is located at 1. A customer with preference θ is located in the spectrum with a distance θ from A and $(1-\theta)$ from O, where $\theta \sim U(0,1)$ is uniformly distributed with unit density on $[0,1]$.[3] The customer's reservation utility, r, is independent of θ and is uniformly distributed on $[R, (1+\lambda)R]$ with a density function $1/(\lambda R)$, where R is the lower bound of customers' reservation utility (in other words, it has to do with the extent to which customers can be exploited). We further assume that R is sufficiently large that OSS is affordable to all customers. In our model, $R \geq 1$ is sufficient to satisfy this condition. Figure 1 shows that the market size is 1 (because $\int_R^{(1+\lambda)R} \int_0^1 (1/\lambda R)\, d\theta\, dr = 1$). For simplicity, we assume that $0 < \lambda \leq 1$. Because $\lambda = \lambda R/R$, the dispersion of reservation utility divided by R, we call λ *relative dispersion*. If $\lambda = 0$, our two-dimensional model would collapse into a one-dimensional model. We summarize the above into the following assumption:

Assumption 4. *Customers are represented by (θ, r), with $\theta \sim U(0,1)$ and $r \sim U(R, R(1+\lambda))$.*

If a customer indexed with (r, θ) adopts A, it gets a net surplus $(r-\theta-p_{At})$ in period t. If he adopts O, then his t-period surplus is $r-(1-\theta)$. If the customer does not adopt any software, his net surplus is zero. A customer's *total net surplus* is the discounted sum of net surplus in two periods. For simplicity, we assume that the discount factor is 1 for all.

There are two periods in the game. In period 1, software vendors simultaneously announce p_{i1}, the

[3] Note that we have normalized on a scale of $[0,1]$. Because we use a two-dimensional model, we can always change scales. For example, if customers' willingness-to-pay is uniformly distributed on $[20, 25]$ and software differentiation is 10 (A locates at 0 and O locates at 10), we can normalize this to 1 and the willingness-to-pay distribution to $U[2, 2.5]$.

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

software price in period 1. Given p_{i1}, customers form rational expectations about the second-period price p_{i2}, then make adoption decisions by comparing the total net surplus from both periods from each software. In period 2, software vendors upgrade their software and charge p_{i2}. A customer may continue to pay for the originally adopted software, stop the subscription, or switch to the other software.

Assumption 5. *A customer will incur a switching cost* s $(s \geq 0)$ *when he switches from the originally adopted software to the other software.*

Customers generally incur costs of reconfiguring hardware, rewriting customized applications, and retraining employees when they switch to a different software. Software vendors upgrade their software in period 2. This increases customers' reservation utility by $v_{up} \geq 0$. If a customer chooses not to upgrade the adopted software, he obtains a basic net surplus of $0 \leq v_b < R$.

Last, we assume that a customer uses at most one software product in each period. Generally, a firm would not use more than one software product to do the same thing, because of cost or efficiency concerns. Under the above assumptions, we analyze the competition between proprietary software and OSS (A versus O).

3. Open Source vs. Proprietary Software

Prior literature assumes that sellers do not commit future prices before customers make adoption decisions. Typically, it is impractical to design a complete contract for proprietary service in period 2. Thus, the price commitment, if any, would not be credible (Fudenberg and Tirole 2002).[4] Software is much different from standardized physical goods because it is even more difficult for a third party (say, the judge) to determine whether the service is really provided at a precommitted price (Shapiro 1986). When customers face high switching costs in period 2, the profit-seeking proprietary software vendor has every incentive to exploit them. This natually casts doubt on the credibility of the software vendor's price commitment, if any.

In contrast to proprietary software, a key feature of OSS is that its commitment of low cost is credible. As defined by the Open Source Initiative (OSI 2011b), open source does not merely mean "free software"; its license terms must satisfy a series of legal criteria (as defined in such open-source license as GPL). First, source code of OSS shall be publicly available. Second, the license of OSS shall not require a royalty or other fees for redistribution. Third, the license of OSS shall allow modifications and redistribution of derived work under the same license terms. As a result, it is difficult for any party to gain monopoly power on an OSS product and then exploit customers after they adopt the OSS. This changes the nature of the game.

Now we proceed to analyze the competition between OSS and proprietary software under the assumptions defined in the previous section. After a customer chooses a software product in period 1, he faces three choices in period 2: (1) upgrade the originally adopted software, (2) not upgrade but keep using the original software, or (3) switch to the other software. Therefore, there are five possible types of customers: AA, OO, AO, OA, and AN, where the first letter stands for the software chosen in period 1 and the second letter stands for the software chosen in period 2.

Accordingly, denote by U_{AA}, U_{OO}, U_{AO}, U_{OA}, and U_{AN} the net surplus of a customer, corresponding to the type as defined above, respectively. The net surplus of type AA customers can be computed as follows:

$$U_{AA} = [r - \theta - p_{A1}] + [r + v_{up} - \theta - p_{A2}],$$

where v_{up} is the incremental reservation utility for the software upgrade. The first term is the net surplus obtained in period 1, and the second term is the net surplus obtained in period 2. Similarly,

$$U_{OO} = [r - (1-\theta)] + [r + v_{up} - (1-\theta)],$$
$$U_{AO} = [r - \theta - p_{A1}] + [r + v_{up} - (1-\theta) - s],$$

noting that the customer incurs a switching cost of s when he switches from A to O (or vice versa) in period 2:

$$U_{OA} = [r - (1-\theta)] + [r + v_{up} - \theta - p_{A2} - s],$$
$$U_{AN} = [r - \theta - p_{A1}] + v_b,$$

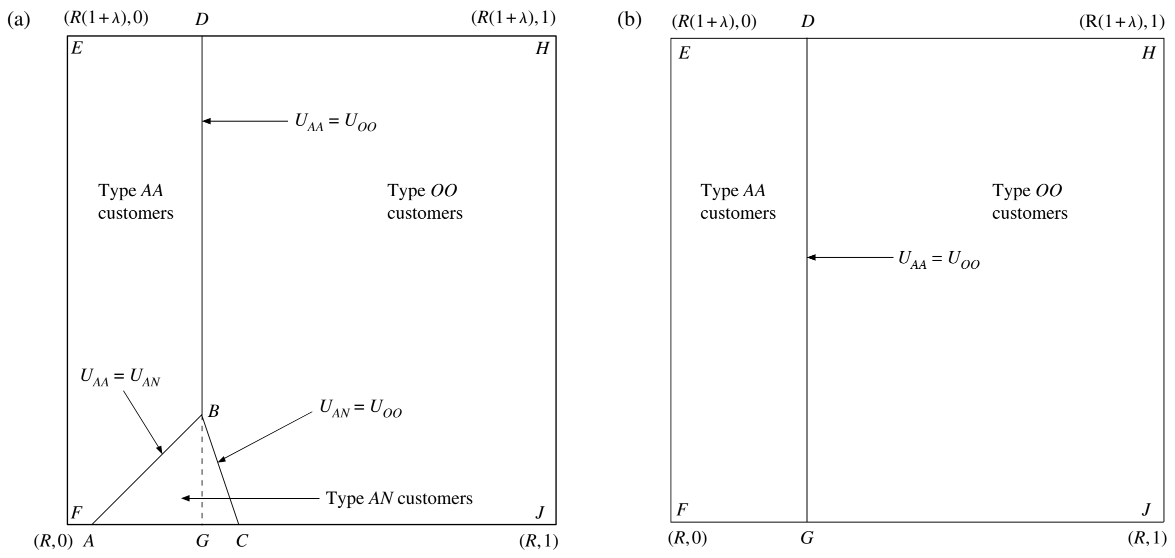
where v_b is the net surplus obtained in period 2 if the customer keeps using the original software without upgrading.

In this section, we focus on the customers who are foresighted. When vendor A announces p_{A1} in period 1, foresighted customers would form a rational expectation of p_{A2}. After comparing their net surplus in both periods from adopting A versus adopting O, customers decide which software to adopt in period 1 and which action to take in period 2. Intuitively, when

[4] For example, a proprietary software provider may commit "lifetime free software upgrade" to its customers. However, after customers adopt its software, the proprietary vendor may sell a "new add-on function" with a high price tag rather than really providing the upgrade service for free.

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

Figure 2 **Segments of Type *AA*, Type *AN*, and Type *OO* Customers**



customers face switching costs in period 2, vendor A may charge a higher price in period 2 than in period 1; that is, $p_{A2} > p_{A1}$. One might expect that if the switching cost is not too high, some customers could switch to O if the price of A becomes too high. Surprisingly, our analytical result shows that such switching behavior will not happen under the optimal pricing of vendor A.

Lemma 1. *In the competition between proprietary software and OSS, under the optimal pricing, no customers will switch to the other software in period* 2.

The key reason for the above is that vendor A will carefully set p_{A1} and p_{A2} such that nobody switches to O in period 2. The intuition is as follows. Foresighted customers know that they will be charged a higher price in period 2. Thus, vendor A needs to set a sufficiently low price to gain customers in period 1. In period 2, if p_{A2} is set so high that some customers of A would switch to O, then these customers must include high-reservation-utility customers[5] with r close to $R(1+\lambda)$. These high-reservation-utility customers are more valuable to vendor A in period 2 than in period 1, because vendor A could charge $p_{A2} \geq p_{A1}$ without losing them. If these high-reservation-utility customers were not valuable to vendor A in period 2, then they should not be valuable in period 1 either. Vendor A should have charged a higher p_{A1} to drop them in period 1 rather than charging a high p_{A2} to drop them in period 2. Thus, vendor A would strategically set p_{A1} and p_{A2} such that the target high-reservation-utility customers would not switch in period 2.

Although switching to the other software does not happen under the optimal pricing of vendor A, it is possible that some customers may choose not to upgrade in period 2. Then two possibilities exist: (1) some of vendor A's customers upgrade and the others keep using A, or (2) all vendor A's customers upgrade in period 2.

Consider the first possibility. There are two kinds of marginal customers in period 1: (1) those who are indifferent between being type AA customers and being type OO customers and (2) those who are indifferent between being type AN customers and being type OO customers. Both kinds of customers use A in period 1. Solving $U_{AA} = U_{OO}$ and $U_{AN} = U_{OO}$ for marginal customers, we obtain $\theta = \frac{1}{4}(2-p_{A1}-p_{A2})$ and $\theta = \frac{1}{3}(2-r-p_{A1}+v_b-v_{up})$. In period 2, type AA customers upgrade, and type AN customers do not. The marginal customers are defined by $U_{AA} = U_{AN}$, or $\theta = r - p_{A2} + v_{up} - v_b$. Figure 2(a) illustrates the three possible regions: type AA customers in Region FABDE, type AN in Region ABC, and type OO customers in Region DBCJH.

A counterintuitive result is that customers in Region ABC would choose A even though they know that they cannot afford p_{A2} in period 2. One might argue that these customers are myopic and irrational. But our result suggests that it could be a rational choice for them, because the subsidy offered by a proprietary software vendor in period 1 can be so attractive to these low-reservation-utility customers that they find it better off just to bite the "marketing bait" than to use O in both periods.

[5] Solving $U_{AA} = U_{AO}$ for marginal customers, we get customers (r, θ) with $\theta \geq \frac{1}{2}(1+s-p_{A2})$ switch to software O. Note that there is no restriction on r. It shows that high-r type customers would also switch to software O if switching to O were ever to happen.

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

Now consider the second possibility, where all vendor A's customers upgrade in period 2 (see Figure 2(b)). The marginal customers in this case are those who are indifferent between being type AA customers and being type OO customers. Solving $U_{AA} = U_{OO}$, we obtain the location of marginal customers: $\theta = \frac{1}{4}(2 - p_{A1} - p_{A2})$.

It can be shown that when $s = 0$, then $p^*_{A1} = p^*_{A2} = \frac{1}{2}$, $\pi^*_A = \frac{1}{4}$, and all vendor A's customers will upgrade in period 2. Next, we use $\pi^*_A(s = 0) = \frac{1}{4}$ as a benchmark for comparison. We obtain the following result:

Lemma 2. *If some of vendor A's customers do not upgrade in period 2, then vendor A would obtain a lower profit than that in an identical market with no switching cost (i.e., $\pi^*_A(s = 0)$), where all of vendor A's customers upgrade in period 2.*

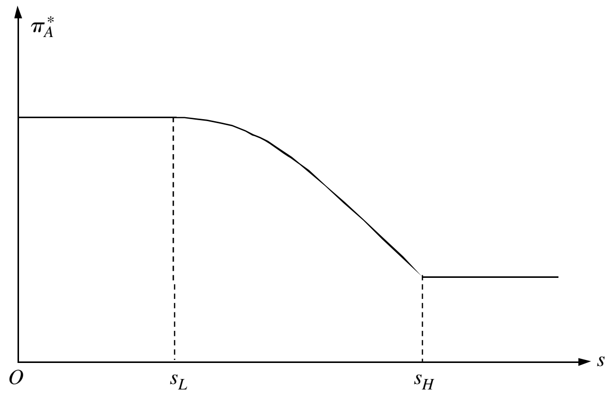
Lemma 2 shows a situation that A would try to avoid. Based on this, we can now give conditions under which vendor A (1) loses some low-reservation-utility customers in period 2 or (2) retains all its customers in period 2.

Proposition 1. *When switching cost is low ($s \le s_L$), the proprietary software vendor obtains the same profit as it would in an identical market with no switching cost. When switching cost is high ($s > s_L$), the vendor obtains a lower profit than it would in an identical market with no switching cost. Here $s_L = R - v_b + v_{up} - \frac{3}{4}$.*

The proof is provided in the appendix. The intuition is as follows. OSS commits its second period price for continuing usage and upgrade. Such a commitment turns out to be O's killer advantage. Rational customers know that if they adopt A in period 1, they will be heavily exploited in period 2 when the switching cost is high. To attract these customers, vendor A needs to set a substantially low p_{A1}. It is true that a higher switching cost enables vendor A to charge a higher price in period 2, yet it also forces vendor A to charge a lower price to sign up customers in period 1. However, a low p_{A1} not only attracts high-reservation-utility customers but also opens the gate for low-reservation-utility customers. Those low-reservation-utility customers will choose not to upgrade in period 2. That is, vendor A captures some "unworthy" customers, who enjoy the subsidy in period 1 but do not bring any profits in period 2. Thus, the existence of low-reservation-utility customers reduces vendor A's profit. Surprisingly, a *higher switching cost makes it more difficult for vendor A to capture its desired customers*.

When the switching cost is sufficiently low, vendor A is unable to exploit its customers in period 2. Customers are comfortable adopting A in period 1 because they do not worry about period 2, when they can always switch to O at a low cost. Thus, vendor A does not need to use a lot of resources to subsidize customers in period 1. This explains why *giving customers the freedom of choice could in fact benefit the proprietary software vendor*.

Figure 3 π^*_A: **A Nonincreasing Function of** s



Notice from $s_L = R - v_b + v_{up} - \frac{3}{4}$ that s_L is a decreasing (increasing) function of v_b (v_{up}). This helps explain how the basic net surplus in period 2 (v_b) and the incremental reservation utility for upgrade (v_{up}) would affect vendor A's profit. Given switching cost (s), a smaller v_b or a larger v_{up} relaxes $s \le s_L$ (that is, shifting s_L to the right in Figure 3). If $s \le s_L$ is satisfied, vendor A may obtain a higher profit than that for $s > s_L$. Thus, vendor A has incentives to reduce v_b or increase v_{up}.

A managerial implication follows. Vendor A may want to stop technical support for nonupgraded software in period 2 such that the customer net surplus obtained from the nonupgraded software (v_b) is low. Then vendor A may offer an attractive upgrade service in period 2 such that customers have high v_{up} for such upgrade in period 2.

Proposition 2. *There exists an s_H such that:* (1) *when $s_L < s < s_H$, vendor A's profit is a decreasing function of s (i.e., $\partial\pi^*_A/\partial s < 0$);* (2) *when $s \ge s_H$, vendor A's profit is not affected by s (i.e., $\partial\pi^*_A/\partial s = 0$).*

If vendor A is able to manipulate the switching cost (s) (say, by increasing the compatibility between A and O), it is possible for that vendor to earn a higher profit by reducing s, when $s_L < s < s_H$. But when $s \ge s_H$, vendor A needs to reduce s to at least below s_H. When $s \le s_L$, reducing switching cost further does not benefit A any more (see Figure 3).

As shown in the proof of Proposition 2, when $s_L < s \le s_H$, vendor A charges p_{A2} in period 2 such that high-reservation-utility customers are indifferent between upgrading A and switching to O; when $s > s_H$, vendor A charges a monopolistic price in period 2 because switching cost is so high that customers are locked in. But vendor A would have to use too many resources to lure customers in period 1,

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

resulting in a overall profit that is actually lower than $\pi_A^*(s)$ with $s < s_H$. Thus, we can conclude:

Corollary 1. *Locking in customers would hurt the proprietary software vendor in the competition against OSS.*

It can be shown that if vendor A were able to credibly precommit its price in period 2, then it would be able to obtain the highest possible profit regardless of the switching cost.[6] If the licensing price of OSS were not zero but a positive value, which would soften the competition, the lock-in strategy is still sub-optimal to vendor A. Note that the proof of Lemma 2 does not rely on the assumption $p_{O1} = p_{O2} = 0$. Using the similar proof, one can easily show that the result of Lemma 2 also holds when the price of OSS is positive.

Prior literature on switching cost has shown that the intensified competition for market share is a key factor that drives the competition between two proprietary products. However, we show in this model that it is the OSS's credible commitment to future price that determines the competition between a proprietary software and an OSS. This illustrates a key difference between these two types of competition. Also, it shows the differentiation between our paper and the prior studies—the models work through different mechanisms, as shown below.

4. Foresighted Customers vs. Myopic Customers

The above analysis assumes that customers are "foresighted" and make adoption decisions based on rational expectations of future payoffs. Yet not all customers behave this way. This is because the payoff of a decision maker who represents the customer is *not* necessarily linked to the total net surplus in two periods. A customer in our model is more likely a company rather than as an individual consumer. For example, a chief information officer works for a company in period 1 only. And his salary is linked to period 1 net surplus of the company, rather than the total net surplus in both periods. We label customers who tend to make decisions based on the current period only as "myopic." We realize that the case of "myopic" customers in itself is not that interesting, but we do this analysis mainly as a comparison on which some insights might be drawn.

A "myopic" customer (r, θ) decides to adopt A or O (or nothing) by comparing $U_A(r, \theta) = \max(0, r - \theta - p_{A1})$ versus $U_O(r, \theta) = \max(0, r - (1 - \theta))$. If $U_A(r, \theta) \geq U_O(r, \theta)$, then it would adopt A; otherwise it would adopt O. We obtain the following result.

[6] This is because vendor A may "copy" the optimal pricing strategy for $s = 0$ by announcing $p_{A1} = \frac{1}{2}$ and precommiting $p_{A2} = \frac{1}{2}$. Then it would obtain $\pi^* = \frac{1}{4}$.

Proposition 3. *Given that customers are "myopic," lock-in will benefit the proprietary software vendor, which will find it optimal to employ the lock-in strategy.*

This is a result opposite to our earlier findings. First, in the competition between A and O, as discussed earlier, OSS' key competitive advantage is its price commitment. However, if customers behave myopically and do not count future payoffs in decision making, such a price commitment is no longer useful. Then it would be less costly for A to compete with O. The above proposition shows that under such a condition, A would find it optimal to employ the lock-in strategy. The above analysis shows that *customer behaviors could significantly alter equilibrium strategy of software vendors*.

5. Broader Effects on Consumers and Social Welfare

So far, we have focused mainly on the effects of lock-in on software vendors. In this section, we look into the broader effects on customers and social welfare. We return to the world of forward-looking customers and focus on the case where *R is sufficiently large* that closed-form solutions can be obtained as follows.

Lemma 3. *When R is sufficiently large and the competition is A versus O, the equilibrium prices and profit are*:

$$p_{A1}^* = \frac{15 + \lambda}{2(6 + \lambda)} - R,$$

$$p_{A2}^* = \frac{3(\lambda - 1)}{2(6 + \lambda)} + R,$$

$$\pi_A^* = \frac{3}{2(6 + \lambda)}. \tag{1}$$

Now we can examine the lock-in effect on an arbitrary customer (r, θ). Such effect can be measured by the change of consumer surplus (CS):

$$\Delta CS(r, \theta) = CS(r, \theta) - CS^N(r, \theta),$$

where $CS(r, \theta)$ is customer (r, θ)'s surplus with lock-in, and $CS^N(r, \theta)$ is its counterpart without lock-in. As shown above, low-reservation-utility customers cannot afford p_{i2} $(i = A, B)$, whereas high-reservation-utility customers can. Considering the effect of lock-in on these two types of customers, we obtain the following result.

Proposition 4. *Lock-in tends to benefit certain low-reservation-utility customers but hurts high-reservation-utility customers.*

Intuition suggests that customers should always refuse lock-in. Proposition 4 confirms this intuition on the one hand, but on the other hand it indicates that lock-in may benefit some customers under certain conditions. This is because lock-in leads to a "bargain-then-rip-off" pricing scheme. The subsidy in period 1 gives low-reservation-utility customers

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

greater surplus in a single period than in two periods combined where no switching costs exist.

Still, it remains unanswered whether lock-in increases the overall social welfare. This question is important for social planners and policy makers. Motivated by these considerations, we examine the lock-in effects on *total* consumer surplus and social welfare and obtain the following result.

Proposition 5. *Lock-in reduces social welfare and total consumer surplus.*

Overall, from a social planner's perspective, lock-in should always be discouraged because lock-in generates social inefficiency for those customers who are locked in but cannot afford the monopolist price in period 2.

6. Closing Remarks

This paper studies the question of how a proprietary software provider competes against not-for-profit OSS. In particular, it examines the incentives for software providers to employ the lock-in strategy and the effects of such a strategy on competitive behavior, customers, and social welfare. In this closing section, we try to pull the results together without being burdened by the mathematical details.

Previous literature has shown that lock-in could hurt or benefit customers, depending on whether the overall competition is intensified (Klemperer 1987b). Several papers have examined the effects of switching cost on competition between two proprietary firms. Our paper differs from the literature in that we consider a new type of competition: the competition between a strategic proprietary software vendor and a not-for-profit OSS, whose commitment to future price is made credible by the open source licensing scheme (such as GPL). Hence, the switching cost does not necessarily intensify the competition in period 1 as would happen in the competition between two proprietary firms.

Our model shows that switching cost hurts, rather than benefits, the proprietary software provider. The key reason is that the OSS can credibly precommit its future price. This would induce the proprietary software provider to change its pricing behavior. That is, it has to use a subsidy to attract customers in the first period; this would capture not only desirable high-reservation-utility customers but also some low-reservation-utility customers. The latter would not generate profits in the postadoption period. Thus, the lock-in strategy is too costly for the proprietary software provider when it competes with OSS. Hence we predict that giving customers freedom of choice could in fact benefit the proprietary software provider.

Our finding is supported by the industry trend of increasing interoperability between OSS and proprietary software. Microsoft launched its "open source project" and announced that it would promote interoperability between its own software and OSS. Oracle is also moving toward compatibility with open source. Existing theories on switching costs do not rationalize such behavior very well, but such moves seem quite rational in the context of our model.

In contrast to popular belief, we show that it is not necessarily true that all customers would be worse off with lock-in. The intensified price competition could benefit some customers (especially those with low-reservation-utility). This is because software vendors use subsidy to attract desirable high-reservation-utility customers in period 1, but such subsidy turns out to be "candy" to low-reservation-utility customers, which would otherwise not be available in a market without lock-in. Farrell and Klemperer (2004) suggest that switching costs generally harms consumers. In contrast, we show that lock-in actually always benefits certain low-reservation-utility customers.

Although lock-in could benefit some customers, it is always socially undesirable when its overall effect is counted in. Its social inefficiency should raise red flags for policy makers. This also helps clarify why governments in many countries are promoting OSS and discouraging "lock-in" by software providers (Evans and Reddy 2003). For example, Microsoft pledged broad support for interoperability and open standards under the pressures of the European Commission (McDougall 2008). The French Parliament passed a law that requires vendors of digital-music players and online music services to open their technical standards and become entirely interoperable. It is widely seen as an attack on Apple, whose iTunes tracks are playable only on its iPod music player. The law, if enacted, will free consumers from digital lock-in.

The above results must be interpreted within the assumptions and limitations of our analytical model. We hope future research will relax some of these assumptions and extend the model. For example, our model has two periods. It might be interesting to examine whether a multiperiod or infinite game would bring additional insights. Intuitively, software vendors might get higher monopoly profits if they lock in adopters for additional periods. However, as we have shown in this paper, greater future profits could also intensify the initial competition for market share. Thus, a greater number of periods could increase or decrease vendors' desire to use the lock-in strategy. We hope that the initial results presented in this paper will motivate more research in this important area.

Copyright: INFORMS holds copyright to this *Articles in Advance* version, which is made available to subscribers. The file may not be posted on any other website, including the author's site. Please send any questions regarding this policy to permissions@informs.org.

7. Notation

θ	Customer's preference for horizontally differentiated software
r	Customer's willingness to pay for a software product
R	Lower bound of customers' willingness to pay
λ	Relative dispersion of customers' willingness to pay
s	Cost of switching to the other software product in period 2
p_{it}	Price of software i ($i = A, O$) in period t ($t = 1, 2$)
v_{up}	Incremental willingness to pay for the upgraded software
v_b	Basic net surplus of a customer who does not upgrade software in period 2
U_{AA}	Net surplus of a customer using software A in period 1 and upgrading it in period 2
U_{AO}	Net surplus of a customer who uses software A in period 1 but switches to software O in period 2
U_{OO}	Net surplus of a customer using software O in two periods
U_{OA}	Net surplus of a customer who uses software O in period 1 but switches to software A in period 2
U_{AN}	Net surplus of a customer using software A in period 1 but not upgrading it in period 2

8. Electronic Companion

An electronic companion to this paper is available as part of the online version that can be found at http://isr.journal.informs.org/.

Acknowledgments

Kevin Zhu acknowledges the financial support of the CAREER Award from the U.S. National Science Foundation (NSF 0654400), the financial support of the National Natural Science Foundation of China (Grant 70728002), and the support of PuJiang Plan #10PJ1431100 of Shanghai Science and Technology Fund. Zach Zhou acknowledges the financial support from the Rady School of Management, University of California, San Diego. Both authors thank the editors and anonymous referees for their comments and suggestions. The paper has benefited from feedback received from conferences and seminars in the field.

References

Bitzer, J. 2004. Commercial versus open source software: The role of product heterogeneity in competition. *Econom. Systems* **28**(4) 369–381.

Caminal, R., C. Matutes. 1990. Endogenous switching costs in a duopoly model. *Internat. J. Indust. Organ.* **8**(3) 353–373.

Casadesus-Masanell, R., P. Ghemawat. 2006. Dynamic mixed duopoly: A model motivated by Linux vs. Windows. *Management Sci.* **52**(7) 1072–1084.

Computer Economics. 2005. Key advantage of open source is not cost savings. *Computer Economics*. http://www.computereconomics.com/article.cfm?id=1043.

Computer World. 2004. On-demand computing survey results. *Computer World*. http://www.computerworld.com/action/article.do?command=printArticleBasic&articleId=94026.

Cremer, H., J.-F. Thisse. 1991. Location models of horizontal differentiation: A special case of vertical differentiation models. *J. Indust. Econom.* **39**(4) 383–390.

Economides, N., E. Katsamakas. 2006. Two-sided competition of proprietary vs. open source technology platforms and the implications for the software industry. *Management Sci.* **52**(7) 1057–1071.

Estes, B. T., O. Maxime. 2003. J2EE vs .NET: The choice depends on your needs. *Computer World*. Accessed 2003, http://www.computerworld.com/s/article/84155/J2EE_vs_.Net_The_choice_depends_on_your_needs.

Evans, D. S., B. J. Reddy. 2003. Government preferences for promoting open-source software: A solution in search of a problem. *Michigan Telecom. Tech. Law Rev.* **9**(2) 313–394.

Farrell, J., P. Klemperer. 2006. Coordination and lock-in: Competition with switching costs and network effects. Working paper, http://papers.ssrn.com/sol3/papers.cfm?abstract_id=917785.

Fudenberg, D., J. Tirole. 2002. *Game Theory*. The MIT Press, Cambridge, MA.

Gartner Research Note. 2003. J2EE and .NET will vie for e-business development efforts. http://gartner.metrostate.edu/research/114600/114609/114609.pdf. Gartner, Stamford, CT.

GNU. 2007. GNU general public license (GPL). http://www.gnu.org/copyleft/gpl.html.

Halloween Documents.1998.http://www.catb.org/~esr/halloween.

McDougall, P. 2008. Microsoft pledges broad support for interoperability, open standards. *Inform. Week*. http://www.informationweek.com/news/windows/showArticle.jhtml?articleID=206801067.

Klemperer, P. 1987a. The competitiveness of markets with switching costs. *RAND J. Econom.* **18**(1) 138–150.

Klemperer, P. 1987b. Markets with consumer switching costs. *Quart. J. Econom.* **102**(2) 375–394.

Kucharik, A. 2003. Vendor lock-in. http://searchopensource.techtarget.com/qna/0,289202,sid39_gci913129,00.html.

Lerner, J., J. Tirole. 2002. Some simple economics of open source. *J. Indust. Econom.* **50**(2) 197–234.

Marinoso, B. G. 2001. Technological incompatibility, endogenous switching costs and lock-in. *J. Indust. Econom.* **49**(3) 281–298.

Mustonen, M. 2002. Copyleft—The economics of Linux and other open source software. *Inform. Econom. Policy* **15**(1) 99–121.

OSI. 2011a. OSI mission: http://www.opensource.org/.

OSI 2011b. OSS definition: http://www.opensource.org/docs/osd.

Roberts, J., I. Hann, S. Slaughter. 2006. Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Sci.* **52**(7) 984–999.

Schmidt, K., M. Schnitzer. 2002. Public subsidies for open source? Some economic policy issues of the software market. Working paper, http://papers.ssrn.com/sol3/papers.cfm?abstract_id=395461.

Shapiro, C. 1986. Investment, moral hazard, and occupational licensing. *Rev. Econom. Stud.* **53**(5) 843–862.

Shapiro, C., H. Varian. 1999. *Information Rules: A Strategic Guide to the Network Economy*. Harvard Business School Press, Boston.

Viswanathan, S. 2005. Competing across technology-differentiated channels: The impact of network externalities and switching costs. *Management Sci.* **51**(3) 483–496.

von Krogh, G., E. von Hippel. 2006. The promise of research on open source software. *Management Sci.* **52**(7) 975–983.

Zhu, K., K. Kraemer, V. Gurbaxani, S. Xu. 2006. Migration to open-standard interorganizational systems: Network effects, switching costs, and path dependency. *MIS Quart.* **30** 515–539.