Perpetual Licensing vs. Subscription of Software: A Theoretical Evaluation

Jie Zhang
College of Business Administration
University of Texas at Arlington, Arlington, TX 76019
jiezhang@uta.edu

Abraham Seidmann
William E. Simon School of Business
University of Rochester, Rochester, NY 14627
Seidmannav@simon.rochester.edu

Abstract

Software vendors have been using the subscription (leasing) model to replace or supplement the traditional perpetual licensing (selling) model. Durable goods theories generally support such a model. This paper looks at a software vendor who can sell (at a posted price) or lease his product and guarantee that subscribers will always have the latest version of the software. We discuss the optimal way of licensing software: perpetual licensing, subscription, or a hybrid approach that involves both. By addressing such specific issues in the packaged software market as network effects, upgrade compatibility, and the vendor’s ability to commit to future prices in a dynamic environment, we demonstrate how a software vendor can manage the trade-offs of perpetual licensing and subscription to achieve a higher profit, as well as the corresponding welfare effect on consumers.

1. Introduction

Software traditionally has been sold as a property. Users pay a fee for a perpetual proprietary license and then can keep using it. A new trend in the software industry is to deliver software products together with maintenance and upgrades to users over the Internet with a subscription model — the so-called Software-as-a-Service (SaaS) model. Many large independent software vendors (ISVs) have followed the trend by adjusting their licensing policies: Microsoft bundled its Enterprise Subscription Agreement for business users with Software Assurance at an annual rate on a three-year term; vendors like Sun, Oracle, SAS Institute, Computer Associates and BMC also offered their own subscription licenses on their major products. Despite the various forms of licensing models, the vendors all aim at generating perpetual revenue streams by transforming this durable good (software) into a subscription-based service — a form of “leasing”.

Industry practitioners claim that this licensing model can lower the cost of ownership and grant users access to up-to-date software at a predictable cost without a large up-front investment. They therefore have maintained that “traditional software is already dead” [9].

Software can be used for a period of time without replacement. In this sense, it is a durable good. Established theories based on conventional durable goods such as books and automobiles ([3], [2], [4], [8]) have raised the time-inconsistency problem in the selling of such goods: If a customer buys a durable good in period 1, he is unlikely to buy it again in future periods. The monopoly seller of the durable good who wants to exploit the residual demand has to lower the price in order to reach lower-value non-adopters. If consumers are rational and patient enough, they will wait to buy until the price drops. Thus [3] conjectured that monopoly sellers of durable goods do not have market power because they are expected to lower their prices in future periods. He suggested that the seller should use leasing to resolve the problem because a lessee does not own the product after each leasing period, and a durable-good monopolist who leases thus has no incentive to cut prices in later periods. [2] formally proved that a durable-good monopolist who leases “can achieve all the standard results of a nondurable monopolist,” so “monopolist sellers do not do as well as renters”.

Hence it seems that both theory and practitioners’ predictions indicate that leasing (subscription) is the dominant strategy for software vendors. Yet before applying that conclusion and moving completely to a subscription model, software vendors still need to consider the following special characteristics of software and evaluate their impacts on licensing decisions:

(1) In general, software cannot be resold or appropriated due to license restrictions, so a second-hand market like that for used cars does not exist;
(2) As an information good, software has a strong economy of scale in production — it is costly to create the first copy, but the marginal cost to produce additional copies is negligible;
(3) It has become relatively easy to improve the value of installed software through upgrades without interfering with the original customization;
(4) The use of software has a strong network externality effect — the value of using any particular software increases with the number of its adopters.
The first three of these special characteristics can be considered merely as a special case of the traditional durable-goods selling or leasing problem. Network externalities ([1], [7]), combined with variable compatibility among different software versions, and different upgrade options, however, complicate users' and vendors' decisions. We therefore cannot be assured that theories regarding the leasing of physical durable goods can be applied directly to the software industry.

In order to provide strategic suggestions to software vendors as well as recommendations to users, we address the following questions by incorporating the special characteristics of software products:

1. What are the benefits and costs of subscription licensing for vendors?
2. Trading off those benefits and costs, what is the optimal way for vendors to license software? Should they use subscription licensing to completely replace the traditional perpetual licensing?
3. How is purchasing behavior influenced by different licensing policies?

Per the page limit in the paper submission requirements, we omit the sections of a dedicated literature review and proofs of Propositions from the complete paper, which are available upon request.

2. Models

We analyze a monopoly software vendor's strategic licensing policy and intertemporal consumer adoption behaviors with a two-period model. Similar to [5], we model the demand for goods with different release dates and varying qualities based on consumers' self-selection. Similar to [4], we assume that the functional form of a software user's net utility is

\[ U(q, x, p; \theta) = \theta q + ex - p, \]  

where \( p \) and \( q \) are the price paid and the quality of the software product, respectively; \( x \) represents the mass of the adopters of the product; and \( \theta \) and \( e \) are the intensity of the quality preference and the network externality effect, respectively. Consumers are heterogeneous in their quality preference \( \theta \), which is uniformly distributed on the support \([0, 1]\), but are homogeneous in their sensitivity to the network effect \( e \).

Software “quality” \( q \) includes such dimensions as speed, compatibility with available operating systems, functionality, user interface, ease of learning, warranty, and other characteristics that affect users' valuation of the product. The monopolist vendor offers a version of the software product in each of the two periods: Version I, with quality \( q_1 \), and Version II, with quality \( q_2 \). Due to the vendor’s continuous research and development effort in improving the quality of its software, the new version software has a higher quality than the previous one \( q_2 > q_1 \).

Consumers discount their future utility gain by a factor of \( \beta \in [0, 1] \). To reduce the number of cases under consideration, we assume that \( (1 + \beta)q_1 \geq \beta q_2 \), so that consumers with a higher quality preference \( \theta \) prefer buying Version I in period 1 as opposed to waiting to buy version II in period 2. This assumption is similar to the one in [5], which only excludes very large improvements between the two versions, but it considerably simplifies the analysis.

Software is usually designed to be compatible with previous versions (backward compatibility) to take advantage of the existing network of users, but forward compatibility (compatibility with future versions) is not easy to achieve. We study the most common case of software production in terms of compatibility: the software is backward compatible but forward incompatible. That is, the later version of the software can successfully use interfaces and data formats from the earlier, but the original software is not designed in such a way that it can seamlessly accommodate files produced with the planned future version. Hence, consumers who upgrade to or buy later versions can enjoy network externalities from the users of both versions, but those who continue using the older version only have a network of users of the same version.

The software vendor can provide the software to the market through one-time sales, subscriptions, and sales of upgrades. Suppose that the marginal production cost of software is zero, which is independent of the licensing policies. Even though a recent study [6] has found that the new SaaS firms who provide subscriptions tend to have higher cost than the traditional vendors who use perpetual licensing, our assumption regarding the marginal cost makes the subscription model more attractive, which increases the robustness of our major conclusions that subscription policy may be inferior to the mixed licensing policy.

Since the two versions have the same marginal production cost and the new version has a higher quality, we have the following conclusion:

Lemma 1: The software vendor is strictly better off not selling version I in period 2.

The software vendor sets the selling price for version I of the software, \( p_1 \); the upgrade price, \( p_u \); and the second-period selling price for version II, \( p_2 \), if he uses the selling policy. If the software vendor adopts a leasing policy, he decides on the rent, \( p_r \), at the beginning of period 1, and the subscribers agree to pay \( p_r \) in each period in order to use the software. This payment mechanism is equivalent to a lump-sum fee at
the beginning plus a free upgrade in the second period. Since the software vendor cannot tell whether the demand for Version II software is from one who already owns the first version or a new adopter, the incentive compatibility condition requires the vendor to keep the upgrade price \( p_u \) no higher than the selling price of version II; that is, \( p_u \leq p_2 \). Otherwise, everyone will choose to buy the new version rather than upgrade.

Figure 1 depicts all the choices available to users. Those who adopt version I in period 1 have the option of upgrading to the new version at a cost of \( p_u \) or continuing to use the same version without any further payment in period 2, but those who lease in the first period will receive an upgrade for no additional charge besides the rent for the period. By Lemma 1, those who are inactive in the first period can skip Version I and purchase Version II directly (“leapfrogging” behavior) or stay inactive in the second period.

Finally, the expected discounted value for a consumer who is inactive in period 1 is
\[
V_i(\theta) = \beta \max \{U(q_2, x_2, p_2; \theta), 0\}. \tag{4}
\]
A consumer who is inactive in the first period can buy the new version of the software in period 2 (IB) or remain inactive (II).

A consumer of type \( \theta \) will choose how to adopt the software by maximizing the total discounted value:
\[
V(\theta) = \max \{V_b(\theta), V_L(\theta), V_I(\theta)\}. \tag{5}
\]
Taking into account consumers’ adoption strategies, the software vendor will decide the market segmentations under each licensing policy and choose an optimal one.

In order to isolate the impacts of the network effect on the users’ and the vendor’s decisions from the impact of the vendor’s ability to commit, we compare the market equilibria under four combinations of market conditions — with and without a network effect, and when the software vendor can and cannot commit to future decisions. Since we allow the intensity of the network effect to be continuous, \( e \geq 0 \), the Coase scenario is only a special case of ours when \( e = 0 \). Specifically, we solve and compare the market equilibria under the pure selling, pure subscription, and hybrid policies when the software vendor is able (Case 1) or unable (Case 2) to commit to future decisions. The results of the two cases are presented in Sub-sections 3.1 and 3.2 below, respectively. In each sub-section, the equilibrium under a market without a network effect is also derived as a special case based on the general one. Sub-section 3.3 compares the results of the cases with and without commitment.

2.1. Case 1 — with commitment

In this benchmark case, the seller can commit to second-period quantities in advance. The monopolist vendor will compare the equilibria under the three licensing policies and choose the one that brings the highest profit.

2.1.1. Pure leasing

Under this policy, the only way for the users to adopt the software is through subscriptions, and they will be able to use the latest version of the software for a rent \( p_r \) in each period. A user with type \( \theta \) has a value
\[
V(\theta) = \max \{V_L(\theta), V_I(\theta)\}; \ V_L(\theta) \text{ and } V_I(\theta) \text{ are defined in Equations (3) and (4) respectively. Since } V_I(\theta) \text{ is increasing in } \theta, \text{ there exists a cut-off value } \theta_L, \text{ such that consumers with } \theta \geq \theta_L \text{ will get positive utility and will lease; the others will stay inactive. The} \]

\[
\]

<table>
<thead>
<tr>
<th>Period 1</th>
<th>Period 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>SV provides Version I</td>
<td>SV provides Version II</td>
</tr>
<tr>
<td>- Buy</td>
<td>- Upgrade to version II</td>
</tr>
<tr>
<td>- Lease version I</td>
<td>- Retain</td>
</tr>
<tr>
<td>- Inactive</td>
<td>- Auto upgrade to version I</td>
</tr>
<tr>
<td></td>
<td>- Buy version II</td>
</tr>
<tr>
<td></td>
<td>- Inactive</td>
</tr>
</tbody>
</table>

Figure 1: Consumer Strategies.
The inverse demand function can be obtained by
\[ V_i(\theta_i) = 0 : \]
\[ p_r = \frac{(q_1 + \beta q_2)\theta_L + e(1 - \theta_L)}{1 + \beta}. \]  
(6)

Within a reasonable degree of the network effect, \(e \leq \frac{q_1 + \beta q_2}{1 + \beta}\), the market is not fully covered in the sense that not everyone in the market will lease.

Based on the users' demand, the vendor maximizes profits over both periods by choosing the optimal cutoff value \(\theta_L^*\):
\[ \max_{\theta_L} \Pi(p_r(\theta_L)) = (1 - \theta_L)(1 + \beta)p_r. \]  
(7)

The maximization problem yields
\[ \theta_L^* = \frac{1}{2} + \frac{(1 + \beta)e}{2(q_1 + \beta q_2 - (1 + \beta)e)}. \]  
(8)

The optimal rent is \(p_r = \frac{q_1 + \beta q_2}{2(1 + \beta)}\). The vendor's profit is
\[ \Pi^L = \frac{(q_1 + \beta q_2)^2}{4(q_1 + \beta q_2 - (1 + \beta)e)}. \]  
(9)

We can observe that the network effect increases the vendor's market share and profit, as well as each user's utility.

### 2.1.2. Pure selling

We consider the case in which the vendor can only sell the software, and he has the ability to commit to future decisions. Thus, the vendor sells Version I to a mass of \(x_0\) users at price \(p_1\) in period 1, and in period 2 he sells Version II to a mass of \((x_2 - x_0)\) users at the full price \(p_2\), and a mass of \((x_0 - x_1)\) early adopters at the upgrade price \(p_u\).

At the beginning of period 1, a consumer of type \(\theta\) evaluates her value of either of the two choices: \(V(\theta) = \max\{V_B(\theta), V_I(\theta)\}\); \(V_B(\theta)\) and \(V_I(\theta)\) are defined in Equations (2) and (4) respectively. There exists a trade-off between buying and waiting: if the consumer buys the software in period 1, she can use it in period 1 and has the option to upgrade in period 2; otherwise, she cannot use the software in period 1, but she retains the option of buying the new version in period 2. Thus, a user has four unique strategies over the two periods: buy version I in period 1 and upgrade to version II in period 2 (BU), buy version I in period 1 and keep using it in period 2 (BH), wait to buy version II in period 2 (leapfrogging) (IB), or remain inactive in both periods (II). By the assumption made above about quality improvement, \((1 + \beta)q_1 \geq \beta q_2\), we have the potential market segmentation as described in Lemma 2. The cutoff values \(\theta^*_0\), \(\theta^*_1\), and \(\theta^*_2\) make consumers indifferent to two adjacent choices.

**Lemma 2**: Consumers with quality preference \(\theta \in [0, \theta^*_0]\) are active; those with \(\theta \in (\theta^*_0, \theta^*_1]\) wait to buy version II in the second period; those with \(\theta \in (\theta^*_1, \theta^*_2]\) buy version I in period 1 but do not upgrade in period 2; and those with \(\theta \in (\theta^*_2, 1]\) always use the latest version of the software during the two periods.

The network effect complicates consumers' decisions: each user internalizes the network externalities caused by other users' decisions into her utility to decide her best strategy. If \(x_i\) \((i = BU, BH, IB,\) and II) represents the demand from each user segment, the inverse demand system is described by the following simultaneous equations:

\[ V_B(\theta_0) = V_{BH}(\theta_0 | p_2, x_2, q_2) \]
\[ V_B(\theta_1 | p_2, x_2, q_2) = V_{BH}(\theta_1 | p_1, x_0, x_1, q_1) \]
\[ V_{BH}(\theta_2 | p_1, x_0, x_1, q_1) = V_{BU}(\theta_2 | p_1, x_0, x_2, q_2) \]
\[ x_2 = 1 - \theta_0 \]
\[ x_0 = 1 - \theta_1 \]
\[ x_1 = \theta_2 - \theta_1 \]
\[ \theta_2 = 1 - x_{BU} \]
\[ \theta_1 = 1 - x_{BH} - x_{BU} \]
\[ \theta_0 = 1 - x_{IB} - x_{BH} - x_{BU} \]

which generates the inverse demand system:

\[ p_r = (q_2 - q_1) - (q_2 - q_1 - e)x_{BU} + ex_{BH} \]
\[ p_1 = (1 + \beta)q_1 - ((1 + \beta)q_1 - \beta)q_2x_{BH} - \beta q_2 x_{BH} \]
\[ p_2 = q_2-q_1-e \]
\[ x_{BU} \geq 0, x_{BH} \geq 0, x_{IB} \geq 0, x_{II} \geq 0 \]
\[ p_2 \geq p_u \]

\[ \max_{x_{BU}, x_{BH}, x_{IB}} \Pi = p_1(x_{BU} + x_{BH}) + \beta(p_u x_{BU} + p_2 x_{IB}) \]  
(10)
To solve this constrained optimization problem, we compare the profits under different combinations of Kuhn-Tucker conditions and find that the optimal solution involves no users' leapfrogging. When the quality improvement of the software is sufficiently large — i.e., \( q_2 > \frac{3 + 2\beta}{2 + \beta} q_1 \) — and the intensity of the network effect is not too large — i.e., \( e < \min\{q_2 - q_1, \frac{2(1 + \beta)q_1^2}{(1 + \beta)q_1 + (2 + \beta)q_2} \} \), the resulting market segmentation is (II, BH, BU), and the masses of adopters are

\[
\begin{align*}
x_{sv}^* &= \frac{1}{2} + \frac{2e(1 + \beta)q_1 - (4 + 3\beta)e^2}{2((4 + 3\beta)e^2 + 4(1 + \beta)(q_1(q_2 - q_1) - eq_2))}, \\
x_{sv}^* &= \frac{e((2 + \beta)q_1 - (3 + 2\beta)q_2)}{(4 + 3\beta)e^2 + 4(1 + \beta)(q_1(q_2 - q_1) - eq_2)}. \\
\end{align*}
\]

The equilibrium prices are obtained by Equation (11).

\[
\Pi^\delta = q_1 + \beta q_2 + (1 + \beta)e + \frac{e^2((4 + 3\beta)(1 + \beta)e + q_1) - (2 + \beta)^2 q_2}{4(4(1 + \beta)(q_1(q_2 - q_1) - eq_2) + (4 + 3\beta)e^2)}.
\]

Given these assumptions and results, we have the following conclusions:

**Proposition 1:** When a software vendor can commit to future decisions and adopts a pure selling policy, with an increase in the intensity of the network effect \( e \),

(i) when \( e \) is not too large, the vendor will reduce the first-period price;

(ii) he will increase the second-period upgrade price;

(iii) when \( e < 2\sqrt{(1 + \beta)q_1(q_2 - q_1)}/(4 + 3\beta) \), there are more consumers buying in period 1 without upgrading in period 2;

(iv) his total market share increases; and

(v) his total profit increases.

When there is no network effect, that is, \( e = 0 \), those users who buy Version I but do not upgrade exit after period 1 and stop contributing to the vendor’s second-period profit. When the vendor has the ability to commit to future decisions, it is optimal for him to increase the price of Version I and lower the upgrade price to convert those consumers who either buy and upgrade or do not buy the software in either period. Thus the market is segmented as (II, BU). The equilibrium market share, vendor profit and consumer surplus are the same as under the pure leasing strategy in §2.1.1.

When the network effect is significant, \( e > 0 \), consumers’ utilities increase with the installed base, so the vendor has a greater incentive to increase the market share in the early period. Those customers who do not upgrade nevertheless will increase the installed base of the new version of the software, and they thus help increase other users’ utilities to the benefit of the vendor. With an increase in the intensity of the network effect, then, the software vendor will lower the first-period price to attract more early buyers. As a result, he can sell Version II to more users at higher selling and upgrade prices. Summing up the discounted profits over the two periods, the vendor is better off with a greater network effect.

When the software vendor can make a commitment, a selling policy provides users with more options in software adoption, including the option to buy Version I without the obligation of upgrading. Those buying-and-owning consumers increase the size of the network that uses the software, which helps the vendor in creating greater positive network externalities, attracting more future users, and then extracting more surplus from the users.

### 2.1.3. Hybrid

Here we consider the mixed strategy of both leasing and selling. Consumers with the greatest preference for quality would like to adopt the latest version of the software. Hence they receive the same value through leasing or buying plus upgrading. When both policies are available, those consumers will choose the one with a lower cost. Consequently, those market segments will not co-exist unless they cost the same, under which condition high-end consumers are indifferent to either strategy. For simplicity and without loss of generality, we only consider consumers’ pure strategy equilibrium and ignore a mixed strategy of either buying or leasing. Thus, for the hybrid case to exist, we need to have

\[
(1 + \beta)p_1 < p_1 + \beta p_2.
\]

Similar to the simultaneous equation in the perpetual licensing case, the inverse demand system is

\[
\begin{align*}
p_x &= (q_1 + \beta q_2)(1 - x_2) - \beta x_{hl}(q_2 - e) + ex_2 - (q_1 - e)x_{hl} \\
p_l &= (1 + \beta)q_1(1 - x_2) - (1 + \beta)(q_2 - e)x_{hl} - \beta q_2 x_{hl} \\
p_2 &= q_2(1 - (x_{hl} + x_{hl} + x_2)) + e(x_{hl} + x_{hl} + x_2).
\end{align*}
\]

Considering the consumers’ selection, the vendor maximizes his discounted total profit by choosing the optimal market segments:

\[
\max_{x_{hl}, x_{hl}, x_2} \Pi = p_x x_{hl}(1 + \beta) + p_l x_{hl} + \beta p_2 x_{hl}.
\]

\[
\text{s.t. } x_{hl} \geq 0, x_{hl} \geq 0, x_2 \geq 0, x_2 \geq 0.
\]

Since the vendor is able to commit in this case, the equilibrium under the hybrid policy leads to the same
profits (Equation (10)) as under the pure selling strategy: \( \Pi^H = \Pi^S \). The market segmentation is similar to the perpetual licensing case; \( x_L \) is replaced by \( x_{BL} \). In this case, therefore, adding the leasing policy does not improve the vendor’s profit over that derived from pure selling. The total discounted price charged to lease and buy-and-upgrade customers should be the same in order to eliminate an arbitrage opportunity: 
\[
p_r^H = \frac{p_r^S + \beta p_u^S}{1 + \beta},
\]
which is increasing in the intensity of the network effect.

Leasing helps the vendor commit to future decisions, but the result above shows that when the vendor can commit in other ways, adding leasing as a licensing policy has no impact on his profit. We next compare the three policies in terms of prices, profits, and consumer surplus and social welfare.

### 2.1.4. Comparison of the three policies

When there is no network externality effect and the vendor can commit to second-period decisions, it is optimal for the vendor to sell only to high-end users and not to buy-and-hold users or the opportunistic consumers who leapfrog. The vendor has no preferred strategy, as all the prices, allocations, and profits will be the same under any of the three.

Software has a strong network effect because there is value created through file sharing and knowledge exchange among adopters, as well as from compatible products ([1]). Users therefore will value a software product with a greater network effect more. When we add a network externality effect to Proposition 2 and compare the market equilibrium, we obtain Proposition 3.

**Proposition 2:** When a software vendor can commit to future decisions and there is a network effect,

(i) a pure selling or hybrid policy weakly dominates a pure subscription strategy for the vendor:
\[
\Pi^S \leq \Pi^H \leq \Pi^H;
\]

(ii) those consumers who buy and do not upgrade incur a lower total expense under the selling policy than under the leasing policy \( p_r < (1 + \beta) p_r \), while those who always use the latest version have to pay more under the selling policy than under the leasing policy \( p_r^H > (1 + \beta) p_r^S \); and

(iii) the consumer surplus and social welfare are no higher when the vendor chooses to lease:
\[
CS^L \leq CS^S = CS^H, \quad W^L \leq W^S = W^H.
\]

We find that the network effect negates the Coase Conjecture that a monopoly seller of a durable good cannot achieve as much profit by selling as by leasing ([2], [3]). When the committed vendor sells the software, he can have both a greater market share (Figure 2) and a higher profit (Figure 3).

![Figure 2. Market Segmentations of Selling or the Hybrid Policy (the dotted curve) vs. Leasing (the solid curve) in Case 1 when \( \beta = 0.7, q_1 = 0.9, q_2 = 2.18 \).](image)

![Figure 3. Profit from Selling or the Hybrid Policy (the dotted curve) vs. Leasing (the solid curve) in Case 1 when \( \beta = 0.7, q_1 = 0.9, q_2 = 2.18 \).](image)

Consumers are also better off under a pure selling or hybrid policy. The selling policy allows the medium-value consumers to buy Version I without the obligation of upgrading, while a pure subscription policy forces them to upgrade by contractual agreement. The purchase option therefore significantly reduces those users’ cost of using the software. With the consequent increase in the population of users, the network externality effect becomes stronger, enabling the vendor to charge a higher upgrade price to the high-value consumers. Consumers who always use the latest version of the software thus incur a higher total cost. Their loss, however, is outweighed by other consumers’ gains, so the total consumer surplus is greater.

We next examine whether these results hold when the vendor cannot commit to future decisions.

### 2.2. Case 2 — no commitment

The above case assumes that the seller is able to commit to his second-period prices. To see the effect of this assumption, we next examine, after period 1, whether the software vendor has an incentive to deviate from his commitment if without a contractual provision.
Then we consider the optimal licensing policy when the seller cannot make a credible commitment to second-period decisions when he sells. Since leasing is a way of committing to future decisions by introducing an external constraint, the vendor’s inability to commit does not affect the equilibrium of the leasing strategy, which is the same as that in §3.1.1. We next solve for the equilibria under the pure selling and hybrid policies and compare them with that under the pure leasing policy.

2.2.1. Pure selling

This is a two-period game. In period 1, the vendor decides the supply of version I of the software $x_B$, and consumers decide whether to buy. In period 2, the vendor decides the price of version II and an upgrade price by choosing the masses of consumers who upgrade, $x_{BU}$, and who buy the new version of the software, $x_{IB}$. Consumers in period 2 choose whether to upgrade, if they buy the first version, or, if they have waited, whether to buy version II or remain inactive. We solve the game backward, starting from the decision in the second period, taking the first period’s outcome as given. In the second period, a consumer who has bought the software may or may not upgrade, depending on her type, the price of the upgrade, and the quality improvement. A consumer who did not buy the software in period 1 will buy the new version in the second period if her net utility is nonnegative.

The consumers’ possible choices are described in §2.1.2. The potential market segmentation is (II, IB, BH, BU). Thus, a consumer’s strategy in the second period depends on her type and the two cutoff values $\theta_2$ and $\theta_0$. The market share affects consumers’ adoption decisions, which in turn result in market segmentation. The inverse demand system is

$$p_u = (q_2 - q_1) - (q_2 - q_1 - e)x_{BU} + e x_{IB}$$
$$p_1 = (1 + \beta)q_1 - (1 + \beta)(q_1 - e)x_0 + e \beta x_{BU} - \beta q_2 x_{IB}$$
$$p_2 = q_2 - (q_2 - e)(x_{IB} + x_0)$$

In period 2, the software vendor takes the first-period equilibrium market share $x_0$ as given and maximizes his second-period profit by choosing the optimal masses of consumers who upgrade and who adopt Version II in period 2:

$$\max_{x_0 > x_{BU}} \Pi' = x_{BU} p_u + x_{IB} p_2$$

s.t. \[ x_{II} \geq 0, x_{IB} \geq 0, x_{BH} \geq 0, x_{BU} \geq 0, \quad (11) \]
$$p_2 - p_u \geq 0$$

Given the mass of adopters of the first version of the software in Case 1, $x_0$, the optimal market segments in period 2 are

$$x_{BU} = \frac{1}{2} + \frac{e(2(3 - x_0)q_2 - e(3 - 2x_0))}{2(4q_2(q_2 - q_1) - 4e(2q_2 - q_1) + 3e^2)}$$
$$x_{IB} = \frac{1 - x_0}{2} + \frac{e(6(q_2 - q_1) - e(3 + x_0))}{2(4q_2(q_2 - q_1) - 4e(2q_2 - q_1) + 3e^2)}$$

The constraints in Equation (11) require $0 \leq x_{IB} \leq 1 - x_0$ and $x_0 \leq x_{BU} \leq \frac{(1 - x_{BU})q_1 - x_{IB}q_2}{q_2 - e}$.

When there is no network effect, i.e., $e = 0$, the vendor will increase the demand from $x_{IB} = 0$ (in the committed case) to $x_{IB} = \frac{1 - x_0}{2}$ by lowering the announced second-period price from $p_2^* = \frac{q_2}{2}$ to $p_2 = \frac{q_2}{4}$. Thus, unless the vendor makes a commitment through contractual provisions or clauses, the optimal decision in the benchmark case is not consistent across time: rational consumers anticipate that prices will fall, which accords with the Coase Conjecture. However, we observe from the market segmentation expressions that when there exists positive network effect and $e$ is large enough to make $(4q_2(q_2 - q_1) - 4e(2q_2 - q_1) + 3e^2)$ negative, it can ameliorate the vendor’s deviation by encouraging him to leverage the installed base $x_0$ and increase his second-period prices $p_2$ (by the inverse demand functions).

Taking the second-period equilibrium into consideration, the seller maximizes his total profit by deciding the number of first-period adopters, $x_0$:

$$\max_{x_0} \Pi = x_0 p_1 + \beta \Pi'$$

s.t. \[ x_0 \geq x_{BU} \]

The network effect gives the vendor an incentive to lower the first-period price to increase his market share and the network externalities. Thus the utilities of future adopters are increased, so that the vendor can raise the upgrade price and selling price in period 2. If the vendor cannot commit to the second-period prices, some medium-value consumers will have the opportunity of waiting to buy version II after the installed base has been built up, and they can benefit from both the network effect and a lower total cost of ownership. In equilibrium, consumers either use the most recent version of the software in both periods or wait and buy the new version in period 2, and the market is segmented as (II, IB, BU). We assume that the intensity of the network effect is not very strong;
i.e., $4q_s(q_s - q_l) - e(7q_s - 4q_l) + 2e^2 > 0$. In equilibrium, the vendor’s optimal first-period market share is

$$x_0 = \frac{1}{2} \cdot \frac{e(5q_s - 2e)}{2(4q_s(q_s - q_l) - e(7q_s - 4q_l) + 2e^2)} \cdot (12)$$

The equilibrium prices can be obtained by the inverse demand functions.

Compared with the equilibrium under the leasing policy, the vendor gains a lower profit because he cannot make a credible commitment. Consumers, however, are better off, as they benefit from more choices and a lower total cost of ownership. The increase in the consumer surplus is even greater than the loss in the software vendor’s profit, so the social welfare is also higher than under the pure leasing strategy.

### 2.2.2. Hybrid

When the vendor offers both selling and leasing options, the potential market segmentation for the hybrid case is the same as that in §2.1.3. Consumers can choose to lease and receive the automatic upgrade in period 2 (type L), buy in period 1 but not upgrade in period 2 (BH), or remain inactive in both periods (II). Their utility functions and the decision problem are described in Equations (2) – (5).

When the network effect is considered, as discussed before, the market share affects consumers’ adoption decisions, which in turn results in market segmentation. The inverse demand system is derived as

$$p_1 = \frac{q_l + q_h}{1 + \beta} - (x_l - (q_l - e)x_{sl} - \beta(q_s - e)x_{sb})$$

$$p_2 = (1 + \beta)(q_l - e)x_l - (1 + \beta)(q_s - e)x_{sh} - \beta(q_s - e)x_{sb}$$

$$p_3 = q_s - (q_l - e)(x_l + x_{sl} + x_{sb})$$

The vendor first solves his second-period profit-maximization problem by choosing the optimal mass of consumers who leapfrog, $x_{sl}$:

$$\max \Pi' = x_Lp_r + x_{sh}p_1 + \beta\Pi'$$

$$s.t. \quad x_L \geq 0, x_{sh} \geq 0, x_{bh} \geq 0, x_L \geq 0$$

The best response function therefore is

$$x_{sh} = \frac{1 - x_{bh} - x_L}{2} \cdot \frac{\beta x_L}{2(1 + \beta)} + \frac{e}{2(q_s - e)} \cdot (14)$$

When the network effect and the quality improvement are large enough, the software vendor may not sell Version II software in period 2; that is, $x_{sh}$ can be 0, and the equilibrium market segmentation can be (II, BH, L). The optimal market shares for that equilibrium are

$$x_L = \frac{(1 + \beta)(q_l^2 - 2e(2q_s - q_l))}{2(q_s - e)((1 + \beta)q_s - q_l - e(1 + 3\beta))}$$

$$x_{bh} = \frac{q_s - 2q_l + 2e(1 + \beta)q_s - (1 + 2\beta)q_l}{2(q_s - e)((1 + \beta)q_s - q_l - e(1 + 3\beta))}$$

The network effect thus may help the vendor commit to not selling Version II in the second period when it is large enough.

The seller maximizes his total profit by deciding the number of first-version lessees, $x_s$, and buyers, $x_{bh}$:

$$\max \Pi = x_Lp_r + x_{bh}p_1 + \beta\Pi'$$

$$s.t. \quad x_L \geq 0, x_{bh} \geq 0, x_{bh} \geq 0$$

When the network effect is large enough, no consumers leapfrog. When there is no network effect, comparing the equilibria under the three policies, we have the following proposition.

**Proposition 3:** In equilibrium, when a software vendor cannot commit to future prices,

(i) when there is no network effect, a pure leasing strategy yields the highest profit to the vendor:

$$\Pi^L > \Pi^S, \Pi^L > \Pi^H$$

(ii) when the network effect is strong enough, a hybrid strategy can yield higher profit to the vendor:

$$\Pi^S < \Pi^L < \Pi^H$$

In this case, offering leases as well as selling helps lock in high-end consumers, so that the vendor can better segment the market and increase prices. Keeping the selling policy, however, still grants the medium-value consumers the option of buying and not upgrading or waiting to adopt the software in the second period. In the absence of network effects, although this hybrid strategy does increase the vendor’s monopoly power compared to the pure selling strategy, leasing is still the dominant strategy for the vendor due to its commitment power through contractual provisions. This result is consistent with the conjecture in [3] and conclusion in [2].

However, the lower prices under the selling strategy benefit the consumers and therefore entice more adopters. The positive externalities created by an enlarged customer base due to the network effect have been ignored by traditional durable-goods theories. We find that when the network effect is strong enough, the profit under the hybrid strategy may be greater than that.
under the pure leasing policy. Since the mathematical expressions are too messy to show, we support part (ii) of Proposition 3 through the following example.

Example: Given $\beta = 0.7$, $q_1 = 0.9$, $q_2 = 2.18$, and $e = 0.42$, the profit under the hybrid policy is greater than either of the pure policies: $\Pi_H = 0.8608 > \Pi_L = 0.8594 > \Pi_S = 0.7468$. We can observe from Figure 4 that when the network effect is large enough, the hybrid strategy dominates. The software vendor gains the highest profit under that strategy, and pure selling generates the lowest profits. Figure 5 shows that the consumer surplus under the hybrid policy is between that under the pure selling policy and that under the pure leasing policy.

A leasing policy has been suggested by [3], [2] and other researchers as a way to commit effectively to second-period prices to resolve the time-inconsistency problem. When we re-examine the Coase Conjecture by considering the network effect, however, we find that leasing may not always achieve the optimal profit for a monopoly seller of durable goods. The intuition is that leasing copes with the time-inconsistency problem of a durable good by treating it as a nondurable good, say, a service. By doing that, leasing causes a discontinuity in the life-cycle of the software. The network effect, however, is path-dependent and allows all the previous owners of the durable good to generate positive externalities. Thus, using leasing to solve the time-inconsistency problem has a side effect of losing those extra network externalities.

Speculations that subscriptions will replace traditional licensing policies have ignored those consumers who only want to use the basic functions of the software product and do not value upgrades very highly. Forcing a transition to pure subscription licensing could drive out those users. The network effect would magnify this loss of market share, which would further reduce the vendor’s profit and the consumer surplus. Our results suggest that a hybrid policy could be a way to remedy this problem.

3. Conclusions

Software vendors now can use the Web to deliver software as a service based on a subscription model. The Coase Conjecture for traditional durable-goods markets, as well as recent speculations of practitioners in the software industry, predict that Web-based subscription is likely to become the dominant means of licensing software.

We, however, find that this type of leasing strategy comes with a cost, which has not attracted much attention in the literature about durable goods. When the good has a strong network effect, as is the case with most software, the subscription model prevents earlier adopters from generating positive externalities to the current users, which reduces the value of the good to consumers and hurts the vendor’s bottom line. On the other hand, perpetual licensing provides users the right to keep using the same version of the software in later periods without further payment. Due to backward compatibility, those users will contribute to the size of the network and therefore add value for later adopters. In the presence of a pronounced network effect, the limitation of leasing outweighs its benefit in creating an ongoing commitment, so that the software vendor is better off to choose a hybrid strategy rather than the pure subscription model.

Our findings reflect the actual licensing practices in the software market. The majority of the software vendors, such as Microsoft, Sun, Oracle and SAS, are not completely switching to the subscription licensing model, but merely have added the leasing channel into their original selling models. Our results provide theoretical support for that strategic decision, as well as suggesting that software vendors should consider the following factors in choosing their optimal licensing strategy:

![Figure 4. Profits of the Hybrid Policy (long dotted curve) vs. Pure Leasing (solid curve) vs. Pure Selling (short dotted curve) in Case 2 and when $\beta = 0.7$, $q_1 = 0.9$, $q_2 = 2.18$.](image1)

![Figure 5. Consumer Surplus of Pure Selling (short dotted curve) vs. Hybrid Policy (long dotted curve) vs. Pure Leasing (solid curve) in Case 2 and when $\beta = 0.7$, $q_1 = 0.9$, $q_2 = 2.18$.](image2)
(1) The anticipated degree of quality improvement: High expected quality improvement will make it more difficult for the vendor to commit to future prices and therefore induce more consumers to await subsequent versions. Vendors should consider using subscription licensing to lock consumers in during the first period.

(2) The intensity of the network externality effect: A strong network effect will increase the vendor’s incentive to lower the price of the first version in order to increase the installed base, thereby increasing the network effect and utilities of future adopters, which will in turn allow the vendor to extract more surpluses from adopters in later periods. When the effect is large enough, vendors should invest in market share and adopt a hybrid policy.

(3) The ability to make a commitment: When software vendors are able to commit through such means as best-price provisions or most-favored-customer clauses, they should prefer a selling strategy, and adding a subscription channel does not increase their profit.

Our findings should also help users realize the benefits and costs of each of the licensing policies and help them make the best decision, taking into account their own characteristics, product upgrades, and network effects. A subscription model smooths out cash payments and replaces a single lump sum with a per period payment, which may be attractive when software budgets are limited. Consumers need to consider the total cost of ownership, however, when evaluating their options. Our model shows that under the pure subscription policy, users can always enjoy the latest version of the software, but they are deprived of the option of not upgrading. The total cost of leasing software is higher than buying but not upgrading. Thus, the claim that the subscription model lowers the cost of ownership is misleading and myopic.

Our consideration of consumer surplus and social welfare likewise has implications for policy makers. The pure leasing policy gives the software vendor more market power but deprives consumers of options. Even though those consumers who always use the latest version software may pay more under the selling policy, their total loss is outweighed by the gain by consumers who choose not to upgrade or who leapfrog under the selling policy. Total consumer surplus therefore is higher under the selling and hybrid policies. Moreover, social welfare is also lowest under leasing according to our Propositions 2 and 3. Hence policy makers should not encourage the pure leasing policy in the software market.

Network externalities combined with compatibility issues endogenize the externality of a user’s adoption decision into the decision of the other users. Thus, network effects significantly complicate the study of software licensing policies. In order to explore the impact of network externalities in particular, we have made several assumptions to simplify the problem. Relaxing those assumptions does not affect our major conclusions but does create interesting opportunities for future research.

References


