Open Source Development with a Commercial Complementary Product or Service

Ernan Haruvy
School of Management
The University of Texas at Dallas
Richardson, Texas, 75083
Phone: (972) 883-4865, Fax (972) 883-6727
Email: eharuvy@utdallas.edu

Suresh P. Sethi
School of Management
The University of Texas at Dallas
Richardson, Texas, 75083
Phone: (972) 883-6245, Fax (972) 883-2089
Email: sethi@utdallas.edu

Jing Zhou
School of Management
The University of Texas at Dallas
Richardson, Texas, 75083
Phone: (972) 883-2382
Email: jingzh@utdallas.edu

March 4, 2005
Abstract

Opening the source code to a software product often implies that consumers would not pay for the software product itself. However, revenues may be generated from complementary products. A software firm may be willing to open the source code to its software if it stands to build a network for its complementary products. The rapid network growth is doubly crucial in open source development, where the users of the firm’s products are also contributors of code that translates to future quality improvements. To determine whether or not to open the source, a software firm must jointly optimize prices for its various products while simultaneously managing its product quality, network size, and employment strategy. Whether or not potential gains in product quality, network size, and labor savings are sufficient to justify opening the source code depends on product and demand characteristics of both the software and the complementary product as well as on the cost and productivity of in-house developers relative to open source contributors. This paper investigates these crucial elements to allow firms to reach the optimal decision in choosing between the open and closed source models.

Keywords: Pricing Research; Optimal control; Open Source; Network Externalities
1 Introduction

The open source development paradigm has proven very successful in recent years in meeting a wide variety of software needs. Open source projects such as Linux, Apache, and Perl have become enormously successful, trampling bigger and better endowed closed source competitors. Their well-publicized success has inspired tens of thousands of open source programmers worldwide and has given a boost to thousands of smaller open source projects (Schiff, 2002). In open source development, volunteers - most often users of the software - collaborate on the development of the software by contributing code and bug reports and by sharing and disseminating the code. Advantages resulting from this form of development include savings on labor costs, increased creativity associated with freedom from company restrictions, and greater development speed arising from rapid dissemination and higher relevance to end user needs (Raymond, 2001). The major caveat is that the code is freely distributed, making it difficult to sell that very same code. Given this characteristic of open source development, a firm wishing to pursue this development model must first carefully formulate its revenue model for the project and weigh the productivity benefits and labor savings against the loss of potential revenue for its software product.

Open source program development may not be appropriate for every firm. The decision to embrace this paradigm and the formulation of related strategies such as price, quality, and hiring critically depend on the business model that is used by the firm to generate revenues for its products. Given that the open source code is free, a firm which coordinates and invests its own resources in an open source project generally has a commercially sold product that would benefit from the open source code - by either nesting the code within its own commercially sold product (this strategy would be consistent with the two-layer design, Cusumano and Selby, 1995), or by exploiting the complementarities that are present between the code and one of its commercially sold products. Although we cannot rule out altruism and social considerations on the part of firms and many firms claim to be motivated by such considerations, we feel it is safe to assume a profit motive in for-profit firms. The public relations aspect of open source sponsorship may be another motive, and the modelling of such benefit is outside the scope of the present work.

Haruvy, Prasad and Sethi (2003) evaluate a model in which the open source code is sold as
part of a commercial product. In that scenario, open source code may become a component of closed source code and the software can be made commercial. This can be done while keeping the source open by providing packaging, distribution, service or brand name or by closing the source, such as Microsoft’s use of the BSD code, or the BSDi operating system (derived from Unix BSD).

In this work we examine a model in a monopoly setting where the open source code is free but complements another product that is sold commercially. The importance of complementarity in software and technology markets with network externalities has been discussed in the economics literature (e.g., Katz and Shapiro, 1985; Shapiro and Varian, 1999; Parker and Van Alstyne, 2004). In recent years, the idea of complementarity has received particular attention in the area of platform competition in two-sided markets, where platform owners are separate from the application developers (Rochet and Tirole, 2003; Armstrong, 2004; Armstrong and Wright, 2004). Economides and Kastamakas (2004) extend this idea to open source and show that application developers would wish to promote open source. Our work is consistent with these ideas. However, our work is not restricted to platforms and is more general. Secondly, in our work the ownership structure of platform and applications is not separate. Lastly, open source is shown to be more than just a free software or platform. We show that a closed source can serve as freeware and indeed our analysis finds that the firm in many instances will keep the source closed yet provide it at no charge to generate faster dissemination of its commercial version. The distinction is that open source is an active tool for improving the quality of the software or platform.

The open source business models discussed in the literature (Raymond, 1999, 2001; Schiff, 2002) suggest several variations on this concept: The Market Positioner model uses the open source code to establish brand name and help the firm’s other products. Netscape’s Mozilla web browser is an example for such a practice. Netscape’s early business model relied upon the sale of server software. The decision to open the source for Mozilla in early 1998 was possibly based on fears that Microsoft would monopolize the browser market, and would eventually drive Netscape out of the server software business (see Raymond, 1999). Another example is Sun’s Star Office which complements Sun’s product lines and brand image. The Compatible Hardware model has an open source software compatible with the firm’s commercially sold hardware. An example is the Apple Mac OS X software which is compatible
with Apple’s hardware. This software is open source since Apple Computer’s decision in mid-March 1999 to open-source “Darwin”, the core of their MacOSX server operating system (for more detail, see http://www.opendarwin.org). The Service and Support model, of which Red Hat is a prominent example, provides service and support for fee to complement the open source code. Finally, under the Information model a firm charges for information associated with the open source code. Consulting services often charge for information relating to open source documentation, maintenance, etc. Notice that the models discussed here can also apply to a freeware software product—a software product which is free but whose code is not necessarily open to the public. However, the key difference lies in the development efforts which are less expensive and more rapid in the open source model. For a discussion of when the freeware model would be most appropriate see Haruvy and Prasad (1998; 2001).

Note that an open source product in the categories discussed here need not be a full software product. It can be a more limited group of modules or classes for existing open source software. For example, IBM initiates and supports various Apache-related projects and releases related code even when it is developed in-house. IBM’s main source of revenue in this case is derived from selling web servers, which benefit greatly from improvements and additions to Apache.

Given the known business models, not every firm should pursue open source as its software model. When a firm decides to pursue a software project, it will generally have an in-house developed prototype and some assessment of the in-house development potential. If the potential contribution from outside programmers is not perceived to be large, obviously the firm should not pursue open source. Furthermore, when the benefit to the firm’s complementary commercial product is small relative to the value of the in-house developed software to the consumers, the firm should keep the software product proprietary and extract as much of the consumer surplus as it can.

We characterize price, quality, and hiring paths for firms under both the open source and closed source models. One interesting finding is that under both open and closed source, the software will initially be free (also known as freeware) if there is no lower bound on the price. Otherwise it will be at its lower bound. In this paper, we assume the lower bound to be zero. That is, if the firm wishes to provide a software product free of charge to boost
sales of another product, it may not need to open the source. Interestingly, if the firm selects to close its source, the higher the initial quality of the complementary good, the longer the software will be free.

The optimal decision on opening the source will depend on several factors, including the importance of user contributions, the wages and effectiveness of in-house developers, and the initial qualities of the products. Though it may be optimal for the firm to close the source code, for finitely lived software, open source improves society’s welfare in terms of both quality and productivity.

The solution approach taken here is optimal control. This approach is ideal here since the firm is able to vary the price of its commercial products from period to period and since related variables (state variables) such as quality and network size vary over time.

Such approach has been often taken in the literature to address dynamic pricing problems (e.g., Chintagunta and Rao, 1996; Gaimon, 1986, 1988, 1989; Gallego and van Ryzin, 1993; Elmaghraby and Keskinocak, 2003), and dynamic quality problems (Carrillo and Gaimon, 2000; Fine, 1986; Fine and Li, 1988; Mukhopadhyay and Kouvelis, 1997; Kouvelis, Mallick and Mukhopadhyay, 1997; Kouvelis and Mukhopadhyay, 1999; Muller and Peles, 1988). The approach presented here allows for dynamic pricing and quality in the presence of network externalities. A network effect, or network externality (Katz and Shapiro, 1985, 1986), is the idea that utility from a product is increasing in the number of other users. It is a pervasive feature of the markets for software and other information goods (Shapiro and Varian, 1999). A network effect implies that the larger the network of existing users the more likely non-adopters are to adopt.

Also note the parallel between the present paper and some works in the innovation diffusion literature. Diffusion is generally defined as the process by which information about an innovation is communicated through a social system (Rogers, 1983). In essence, diffusion can be thought of a special type of network effect which, when accounting for the dependence between demand and price, translates to a dynamic pricing problem for a product (e.g., Kalish, 1983; Kalish and Lilien, 1983; Nascimento and Vanhonacker, 1988; Sethi and Bass, 2003).

The approach we take here expands on the above dynamic pricing literature by examining a scenario of two complementary goods, one of which can be developed through user
contributions contingent on it being free. The paper is organized as follows: Section 2 lays out the basic model and assumptions. Section 3 develops the theory and its analytical implications as they relate to dynamics in finite horizon setting. Section 4 pursues numerical simulations for comparative statics and further insights. Section 5 concludes. The appendix contains all proofs not included in the text.

2 The models and setting

The firm has two products, product 1 and product 2, which have different co-dependent demand rates, $D_1$ and $D_2$, respectively. In the case of closed source development, all quality improvements arise in-house as a function of the number of in-house developers (e.g., Cohen, Eliashberg and Ho, 1996; Joglekar, Yassine, Eppinger, and Whitney, 2001). In the case of open source development, all quality improvements come from the users, as a function of the size $m$ of the network.

$Q_1(t) =$ The quality of the software at time $t$.

$Q_2(t) =$ The exogenously given quality of the complementary product at time $t$.

$m(t) =$ Size of network of users (i.e., installed base) at time $t$.

$N(t) =$ Number of in-house developers at time $t$.

$P_1(t) =$ The price of the software at time $t$.

$P_2(t) =$ The price of the complementary product at time $t$.

The profit in a given period is the revenue $P_1D_1$ from the software, plus the revenue $P_2D_2$ from the complementary product, minus development costs if applicable. In the open source case, $P_1$ is by definition zero, and so the revenue is only $P_2D_2$. This loss of revenue may be offset partly by the fact that development costs are zero. In the closed source case, development costs increase with the number of in-house developers, $N$. We assume a cost function of $wN^2$ in accordance with the economic principle of increasing marginal cost. $w$ is not unit wage but rather a cost parameter. One can think of a generalized cost function $wN^\gamma$. Given the law of increasing marginal costs, $\gamma = 2$ is a reasonable example. In the closed source case, the in-house contributors improve the quality of the software over time. That is,

$$\dot{Q}_1 = kN - \delta Q_1, \quad Q_1(0) = Q_1^0 \geq 0. \quad (1)$$
The parameter \( k > 0 \) denotes the productivity or effectiveness of the in-house closed source programmers. When \( k \) is high, closed source programmers are very effective and open source development may not be warranted. It is assumed that the quality depreciates over time at a constant proportional rate \( 0 < \delta < 1 \). This depreciation can be thought of as obsolescence relative to exogenous technological progress. It is assumed that initial quality of the software is nonnegative. Since the source is closed to the public, contributions by users are not possible. In the open source case, on the other hand, there are no in-house developers and all contribution is through users. Hence,

\[
\dot{Q}_1 = \alpha m - \delta Q_1, \quad Q_1(0) = Q_1^0 \geq 0.
\]  

The parameter \( \alpha > 0 \) represents the level of involvement by the open source user community, which includes users of both the software and of the complementary product.

In addition to software quality \( Q_1(t) \), we also consider the quality of the complementary product \( Q_2(t) \). We allow \( Q_2(t) \) to be a dynamic variable as long as it is nonnegative. We assume, however, that \( Q_2(t) \) is exogenous to this problem, although it may be endogenous to the firm through variables not considered here.

The size \( m \) of the network of users increases each period by the number of new users of both products and decreases by a percentage of the existing users discontinuing the use of the product. The users of both products need not be equally weighted. A user of the software may be more or less valuable to the network than a user of the complementary product. The parameter \( a > 0 \) measures this relative weight. We also assume separability between demands for the two products in the network, though the two are complements. That is, a user who uses both products has the weight of \( (1 + a) \) in the network. The parameter \( 0 < \varepsilon < 1 \) is the rate of depreciation of the network or the rate of exit. Note that the network growth rate is derived from a summation of two implicit separate networks, \( m_1 \) and \( m_2 \), where \( m_1 \) depends on \( D_1 \) and \( m_2 \) depends on \( D_2 \). Each network could conceivably have a different exit rate, denoted \( \varepsilon_1 \) and \( \varepsilon_2 \) respectively. However, for ease of exposition we assume the exit rates to be the same, resulting in a single \( \varepsilon \). Hence,

\[
\dot{m} = aD_1 + D_2 - \varepsilon m, \quad m(0) = m^0 \geq 0.
\]  

The demand of the complementary product is \( D_2 = h(P_2, m, Q_2) \), where \( h(P_2, m, Q_2) \geq 0 \), \( \frac{\partial h}{\partial P_2} \leq 0 \), \( \frac{\partial h}{\partial m} \geq 0 \) and \( \frac{\partial h}{\partial Q_2} \geq 0 \). This means that \( D_2 \) decreases as the price of the
complementary product $P_2$ increases, and increases as the size $m$ of the network of users and the quality of the complementary product $Q_2$ increase.

We assume that the demand of the software is $D_1 = D_2g(P_1, Q_1) = h(P_2, m, Q_2)g(P_1, Q_1)$, where $g(P_1, Q_1) \geq 0$, $\frac{\partial g}{\partial P_1} \leq 0$ and $\frac{\partial g}{\partial Q_1} \geq 0$. This means that $D_1$ decreases as the price of the software $P_1$ increases, and increases as the quality of the software $Q_1$ increases. In the open source case, since $P_1 = 0$, then $D_1 = D_2g(0, Q_1)$. The multiplication of the function $g(P_1, Q_1)$ by $D_2$ represents the effect of complementarity. That is, the demand for the software product increases with the demand for the complementary product. At this point some more discussion is warranted on the relationship between $D_1$ (demand for software) and $D_2$ (demand for the complement). $D_1$ and $D_2$ affect each other, but in different ways. $D_1$ is affected by $D_2$ directly, through complementarities. That is, a portion of the complement’s users ($D_2$) may wish to use the software ($D_1$). However, the software users can use the software independently of the complementary good (they could substitute any number of other products). $D_2$ is affected by $D_1$ indirectly, through the network size (which is a function of purchases of both 1 and 2). $D_2$ is affected by the network because of network externalities. That is, it is affected directly by the size of the network. People derive utilities from other people using the product, though newsgroups, file sharing, increased service and compatible goods. $D_1$ is not affected by the network directly. However, in the open source case, $D_1$ is affected by the network indirectly through the impact of the network on the quality of the code, $Q_1$.

Suppose the products, both the software and the complementary product, have finite lives with a known terminal period $T$ ($0 < T < \infty$). The terminal period could be due to an anticipated release of a new generation of products or technologies or due to a known date for the firm to cease operations. We assume the salvage value of both products is $\sigma(Q_1(T), m(T))$, which is a function of the ending state variables at time $T$. We assume that $\frac{\partial \sigma}{\partial Q_1} \geq 0$ and $\frac{\partial \sigma}{\partial m} \geq 0$.

We summarize the two models as follows:
The firm maximizes its discounted (discount rate is \( \rho > 0 \)) profit stream over the time interval \( 0 \leq t \leq T \), plus the discounted salvage value of both products at time \( T \). The instantaneous profit rate consists of the price for the commercial product (the price for the open source software product, \( P_1 \), is zero) multiplied by the demand for the commercial product at each period.

\[
V_0(0, Q_1(0), m(0)) = \max \left\{ J(P_2(\cdot)) = \int_0^T P_2 h(P_2, m, Q_2) e^{-\rho t} dt + \sigma(Q_1(T), m(T)) e^{-\rho T} \right\}
\]

s.t.
\[
\begin{align*}
\dot{Q}_1 &= \alpha m - \delta Q_1, \quad Q_1(0) = Q_1^0, \\
\dot{m} &= a D_1 + D_2 - \varepsilon m, \quad m(0) = m^0
\end{align*}
\]

where
\[
D_1 = D_2 g(0, Q_1) \\
D_2 = h(P_2, m, Q_2)
\]

In this problem, \( P_2(t) \) is the control variable at time \( t \). \( J(P_2(\cdot)) \) is the objective function. \( V_0(0, Q_1(0), m(0)) \) is the value function given that we start at time 0 in state \( Q_1(0) \) and \( m(0) \). We assume that \( F_0 \equiv P_2 h(P_2, m, Q_2) \) is concave in \( P_2 \). This assumption holds for two...
common demand functions we use later in the paper. We form the Lagrangian:

\[ L = P_2 h(P_2, m, Q_2) + \lambda (\alpha m - \delta Q_1) \]
\[ + \mu [ah(P_2, m, Q_2)g(0, Q_1) + h(P_2, m, Q_2) - \varepsilon m] + \eta_2 P_2. \]  
(8)

From this we get the adjoint equations

\[ \dot{\lambda} = \rho \lambda - \frac{\partial L}{\partial Q_1} = (\rho + \delta) \lambda - a \mu h(P_2, m, Q_2) \frac{\partial g}{\partial Q_1}, \lambda(T) = \frac{\partial \sigma}{\partial Q_1} \bigg|_T, \]  
(9)

\[ \dot{\mu} = \rho \mu - \frac{\partial L}{\partial m} = (\rho + \varepsilon) \mu - \alpha \lambda - [P_2 + \mu (ag(0, Q_1) + 1)] \frac{\partial h}{\partial m}, \mu(T) = \frac{\partial \sigma}{\partial m} \bigg|_T. \]  
(10)

The optimal control must satisfy

\[ \frac{\partial L}{\partial P_2} = h(P_2, m, Q_2) + [P_2 + \mu (ag(0, Q_1) + 1)] \frac{\partial h}{\partial P_2} + \eta_2 = 0, \]  
(11)

and the Lagrange multiplier \( \eta_2 \) must satisfy the complementary slackness condition

\[ \eta_2 \geq 0, \; \eta_2 P_2 = 0. \]  
(12)

**Proposition 1.** In the open source case, \( Q_1(t) \geq 0 \) and \( m(t) \geq 0 \) for \( 0 \leq t \leq T \). Moreover, these variables are strictly positive if their initial values \( Q_1^0 > 0 \) and \( m^0 > 0 \).

**Proposition 2.** In the open source case, the optimal profit \( (a) \) increases with \( \alpha \); \( (b) \) increases with the initial software quality \( Q_1^0 \).

The variable \( \lambda \) is interpreted as the per unit change in the value function for small changes in the software quality \( Q_1 \). In other words, \( \lambda(t) \) is the marginal value per unit of software quality at time \( t \). Similarly, the variable \( \mu \) is interpreted as the per unit change in the value function for small changes in the size of the network \( m \). That is, \( \mu(t) \) is the marginal value per unit of the network at time \( t \). The next three propositions show that \( \lambda \) and \( \mu \) are nonnegative and \( \lambda \) is decreasing over time. These propositions are important because they show that the firm would benefit from higher software quality and network size.

**Proposition 3.** In the open source model, the marginal benefit \( \lambda \) of increasing the software quality is nonnegative.

**Proposition 4.** In the open source model, the marginal benefit \( \mu \) of increasing the size of the network is nonnegative.
Proposition 5. In the open source model, the adjoint variable $\lambda$ decreases over time if the salvage value of the software quality is zero at time $T$.

We next turn to examining prices. We make a distinction between myopic prices, $\hat{P}_2(t)$, and forward looking prices, $P_2^*(t)$. Specifically, myopic prices maximize immediate returns without future consideration, whereas forward looking prices maximize returns over the entire horizon. We show that myopic prices are always excessive relative to forward looking prices. Recall that $F_O(t) = P_2(t) h(P_2(t), m(t), Q_2(t))$ is the revenue accrued at time $t$. Let $\hat{P}_2(m(t), Q_2(t)) = \arg \max_{P_2(t)} F_O(t)$. $P_2^*(t)$ is the solution to (4). Let $m^*(t)$ be the optimal network size corresponding to $P_2^*(t)$ at time $t$.

Proposition 6. $P_2^*(t) \leq \hat{P}_2(m^*(t), Q_2(t))$ for $0 \leq t \leq T$. Moreover $P_2^*(T) = \hat{P}_2(m^*(T), Q_2(T))$ if the salvage value is zero at time $T$.

Proposition 6 is important for several reasons. First, it implies that myopic behavior results in excessive prices. Second, it allows us later on to arrive at an upper bound for the commercial product’s price.

3.2 Closed Source

The firm maximizes its discounted profit stream over the time interval $0 \leq t \leq T$, plus the discounted salvage value of both products at time $T$. The instantaneous profit rate is the revenue of the software and the commercial product minus the development costs at each period. Note that the size of the workforce, $N$, is now a control variable and not a state variable, as $m$ in the open source case. This latter model is consistent with the extant staffing control literature (Gaimon, 1997).

\[
V_C(0, Q_1(0), m(0)) = \max \{J(P_1(\cdot), P_2(\cdot), N(\cdot)) = \int_0^T \left[ P_1 h(P_2, m, Q_2) g(P_1, Q_1) + P_2 h(P_2, m, Q_2) - wN^2 e^{-\rho t} dt + \sigma(Q_1(T), m(T)) e^{-\rho T} \right] - \delta Q_1, Q_1(0) = Q_1^0, \]
\]
\[\dot{Q}_1 = kN - \delta Q_1, \]
\[m = ah(P_2, m, Q_2) g(0, Q_1) + h(P_2, m, Q_2) - \epsilon m, m(0) = m^0, \]
\[P_1, P_2 \geq 0, N \geq 0. \]
In this problem, \( P_1(t) \), \( P_2(t) \) and \( N(t) \) are the control variables at time \( t \). \( J(P_1(\cdot), P_2(\cdot), N(\cdot)) \) is the objective function. \( V_C(0, Q_1(0), m(0)) \) is the value function given that we start at time 0 in state \( Q_1(0) \) and \( m(0) \). We assume that \( F_C \equiv P_1 h(P_2, m, Q_2) g(P_1, Q_1) + P_2 h(P_2, m, Q_2) - wN^2 \) is jointly concave in \( P_1, P_2 \) and \( N \). We form the Lagrangian:

\[
L = P_1 h(P_2, m, Q_2) g(P_1, Q_1) + P_2 h(P_2, m, Q_2) - wN^2 + \lambda(kN - \delta Q_1) + \mu[a h(P_2, m, Q_2) g(P_1, Q_1) + h(P_2, m, Q_2) - \varepsilon m] + \eta_1 P_1 + \eta_2 P_2 + \eta_3 N. \tag{17}
\]

From this we get the adjoint equations

\[
\dot{\lambda} = (\rho + \delta)\lambda - (P_1 + a\mu) h(P_2, m, Q_2) \frac{\partial g}{\partial Q_1}, \quad \lambda(T) = \frac{\partial \sigma}{\partial Q_1} \bigg|_T, \tag{18}
\]

\[
\dot{\mu} = (\rho + \varepsilon)\mu - [P_2 + \mu + (P_1 + a\mu) g(P_1, Q_1)] \frac{\partial h}{\partial m}, \quad \mu(T) = \frac{\partial \sigma}{\partial m} \bigg|_T. \tag{19}
\]

The optimal control must satisfy

\[
\frac{\partial L}{\partial P_1} = \left[ g(P_1, Q_1) + (P_1 + a\mu) \frac{\partial g}{\partial P_1} \right] h(P_2, m, Q_2) + \eta_1 = 0, \tag{20}
\]

\[
\frac{\partial L}{\partial P_2} = h(P_2, m, Q_2) + [P_2 + \mu + (P_1 + a\mu) g(P_1, Q_1)] \frac{\partial h}{\partial P_2} + \eta_2 = 0, \tag{21}
\]

\[
\frac{\partial L}{\partial N} = -2wN + k\lambda + \eta_3 = 0, \tag{22}
\]

and the Lagrange multiplier \( \eta_1, \eta_2 \) and \( \eta_3 \) must satisfy the complementary slackness condition

\[
\eta_1 \geq 0, \quad \eta_1 P_1 = 0, \tag{23}
\]

\[
\eta_2 \geq 0, \quad \eta_2 P_2 = 0, \tag{24}
\]

\[
\eta_3 \geq 0, \quad \eta_3 N = 0. \tag{25}
\]

**Proposition 7.** In the closed source case, \( Q_1(t) \geq 0 \) and \( m(t) \geq 0 \) for \( 0 \leq t \leq T \). Moreover, these variables are strictly positive if their initial values \( Q_1^0 > 0 \) and \( m^0 > 0 \).

**Proposition 8.** In the closed source case, the optimal profit (a) increases with \( k \); (b) increases with the initial software quality \( Q_1^0 \); (c) decreases with \( w \).

The next two propositions show that the firm benefits from higher software quality and higher network size.
Proposition 9. In the closed source model, the marginal benefit $\lambda$ of increasing the software quality is nonnegative.

Proposition 10. In the closed source model, the marginal benefit $\mu$ of increasing the size of the network is nonnegative.

Proposition 11. In the closed source model, the adjoint variable $\lambda$ decreases over time if the salvage value of the software quality is zero at time $T$. Moreover, the number of in-house developers $N$ declines over time and eventually $N$ goes to zero at $T$.

Corollary 1. In the closed source model, the quality of software $Q_1$ increases at first and later decreases.

As in the open source analysis, we make a distinction between myopic prices, $\hat{P}_1(t)$ and $\hat{P}_2(t)$, and forward looking prices, $P_1^\ast(t)$ and $P_2^\ast(t)$, where myopic prices maximize immediate returns without future consideration and forward looking prices maximize returns over the entire horizon. We again show that myopic prices are always excessive relative to forward looking prices. Recall that $F_C(t) = P_1(t)h(P_2(t), m(t), Q_2(t))g(P_1(t), Q_1(t)) + P_2(t)h(P_2(t), m(t), Q_2(t)) - wN(t)^2$ is the profit accrued at time $t$. Let $\left(\hat{P}_1(m(t), Q_1(t), Q_2(t)), \hat{P}_2(m(t), Q_1(t), Q_2(t))\right) = \arg\max_{P_1(t), P_2(t)} F_C(t)$. $(P_1^\ast(t), P_2^\ast(t))$ is the solution to (13). Let $m^\ast(t)$ and $Q_1^\ast(t)$ be the optimal network size and software quality corresponding to $(P_1^\ast(t), P_2^\ast(t))$ at time $t$.

Proposition 12. $P_1^\ast(t) \leq \hat{P}_1(m^\ast(t), Q_1^\ast(t), Q_2(t))$ and $P_2^\ast(t) \leq \hat{P}_2(m^\ast(t), Q_1^\ast(t), Q_2(t))$ for $0 \leq t \leq T$. Moreover $P_1^\ast(T) = \hat{P}_1(m^\ast(T), Q_1^\ast(T), Q_2(T))$ and $P_2^\ast(T) = \hat{P}_2(m^\ast(T), Q_1^\ast(T), Q_2(T))$ if the salvage value is zero at time $T$.

4 Numerical Analysis

We examine different scenarios numerically, in both open source and closed source models. We assume $\sigma(Q_1(T), m(T)) = 0$. Therefore, $\lambda(T) = \mu(T) = 0$. To simplify the calculation, we assume $Q_2$ is positive and constant for $0 \leq t \leq T$.

We examine two types of demand functions. The first one is the negative exponential demand function, which has several desirable properties (see Greenhut and Greenhut, 1977; Anderson, 1989; Haruvy, Prasad, and Sethi, 2003). This functional form has the nice
property that it precludes the possibility of negative demand. The demand function of the complementary product is

\[ D_2 = h(P_2, m, Q_2) = \exp\left(-\frac{P_2}{mQ_2}\right), \] (26)

and the demand of the software is

\[ D_1 = D_2g(P_1, Q_1) = \exp\left(-\frac{P_2}{mQ_2}\right)\exp\left(-\frac{P_1 + c}{Q_1}\right). \] (27)

Since the price \( P_1 \) for the open source software product is zero, in the open source case the demand of the software is

\[ D_1 = D_2g(0, Q_1) = \exp(-\frac{P_2}{mQ_2})\exp(-\frac{c}{Q_1}). \] (28)

The positive constant \( c \) in equation (27) and (28) is necessary to keep the numerator strictly above zero in the open source case. Without the positive constant \( c \), demand \( D_1 \) would equal \( D_2 \) in the open source case.

The second demand function we examine is linear in price. The demand function of the complementary product is

\[ D_2 = h(P_2, m, Q_2) = B - \frac{P_2}{mQ_2}, B - \frac{P_2}{mQ_2} \geq 0, \] (29)

and the demand of the software is

\[ D_1 = D_2g(P_1, Q_1) = (B - \frac{P_2}{mQ_2})(A - \frac{P_1 + c}{Q_1}), (A - \frac{P_1 + c}{Q_1}) \geq 0. \] (30)

In the open source case the demand of the software is

\[ D_1 = D_2g(0, Q_1) = (B - \frac{P_2}{mQ_2})(A - \frac{c}{Q_1}), (A - \frac{c}{Q_1}) \geq 0. \] (31)

We use Fortran code and Excel spreadsheets to find numerical solutions. Due to the computational limitations, numerical solutions are the result of approximations (e.g., Sethi and Thompson, 2000, pp. 48-50). Specifically, for each state and adjoint variable \( x \), we substitute \( \frac{\Delta x}{\Delta t} = \frac{x(t + \Delta t) - x(t)}{\Delta t} \) for \( \dot{x} \). We let \( \Delta t = 0.01 \). As a numerical example, we consider the parameters: \( \delta = 0.03, \varepsilon = 0.01, \rho = 0.1, a = c = k = w = \alpha = 1, Q_1^0 = m^0 = Q_2^0 = 1 \) and \( T = 50 \). These parameter values will be used throughout this section in all the illustrations that follow. Note also that by Proposition 1 and 7, \( Q_1(t) > 0 \) and \( m(t) > 0 \), for \( 0 \leq t \leq T \).
By Proposition 6 and 12, we know that prices have upper bounds defined by the myopic prices. The next two propositions provide the upper bounds for both exponential demand function and linear-price demand function. Figure 1 shows the prices and their bounds.

<Insert Figure 1 here>

To make our presentation easy, we define \( \hat{P}_1^*(t) \equiv \hat{P}_1(m^*(t), Q_1^*(t), Q_2(t)) \) and \( \hat{P}_2^*(t) \equiv \hat{P}_2(m^*(t), Q_1^*(t), Q_2(t)) \).

**Proposition 13.** (i) In the open source model with exponential demand function, the optimal price of the complementary product \( P_2^*(t) \leq \hat{P}_2^*(t) = m^*(t)Q_2(t) \).

(ii) For the linear-price demand function, the optimal price of the complementary product \( P_2^*(t) \leq \hat{P}_2^*(t) = \frac{1}{2}Bm^*(t)Q_2(t) \).

**Proof.** Result follows immediately from Proposition 6 and the demand functions of \( h(P_2, m, Q_2) \). □

The above proposition gives us a "soft" upper bound on price, where "soft" refers to the fact that it is not a prior. However, \( m \) itself has an upper bound, which results in a "hard" upper bound for \( P_2 \), where "hard" refers to independence from the trajectories for state and control variables.

**Corollary 2.** (i) In the open source model with exponential demand function, the optimal price of the complementary product \( P_2^*(t) \) has a hard upper bound \( \tilde{m}(t)Q_2(t) \), where \( \tilde{m}(t) = m^0e^{-\varepsilon t} + (a + 1)(1 - e^{-\varepsilon t})/\varepsilon \);

(ii) For the linear-price demand function, the optimal price of the complementary product \( P_2^*(t) \) has a hard upper bound \( \frac{1}{2}B\tilde{m}(t)Q_2(t) \), where \( \tilde{m}(t) = m^0e^{-\varepsilon t} + (aA + 1)B(1 - e^{-\varepsilon t})/\varepsilon \).

**Proposition 14.** (i) In the closed source model with exponential demand function, the optimal price of the software \( P_1^*(t) \leq \hat{P}_1^*(t) = Q_1^*(t) \). The optimal price of the complementary product \( P_2^*(t) \leq \hat{P}_2^*(t) = m^*(t)Q_2(t) - \hat{P}_1^*(t)g(\hat{P}_1^*(t), Q_1^*(t)) \).

(ii) For the linear-price demand function, the optimal price of the software \( P_1^*(t) \leq \hat{P}_1^*(t) = \frac{1}{2}(AQ_1^*(t) - c) \). The optimal price of the complementary product \( P_2^*(t) \leq \hat{P}_2^*(t) = \frac{1}{2}(Bm^*(t)Q_2(t) - \hat{P}_1^*(t)g(\hat{P}_1^*(t), Q_1^*(t)) \).
**Proof.** Result follows immediately from Proposition 12 and the demand functions of $h(P_2, m, Q_2)$ and $g(P_1, Q_1).$ \[\square\]

**Corollary 3.** (i) In the closed source model with exponential demand function, the optimal price of the software $P_1^*(t)$ has a hard upper bound $\bar{Q}_1(t),$ where $\bar{Q}_1(t) = Q_0^e e^{-\varepsilon t} + kN(0)(1 - e^{-\varepsilon t})/\delta,$ and $N(0)$ is the number of in-house programmers at time 0. The optimal price of the complementary product $P_2^*(t)$ has a hard upper bound $\bar{m}(t)Q_2(t),$ where $\bar{m}(t) = m_0 e^{-\varepsilon t} + (a + 1)(1 - e^{-\varepsilon t})/\varepsilon.$

(ii) For the linear-price demand function, the optimal price of the software $P_1^*(t)$ has a hard upper bound $\frac{1}{2}(A\bar{Q}_1(t) - c),$ where $\bar{Q}_1(t) = Q_0^e e^{-\varepsilon t} + kN(0)(1 - e^{-\varepsilon t})/\delta,$ and $N(0)$ is the number of in-house programmers at time 0. The optimal price of the complementary product $P_2^*(t)$ has a hard upper bound $\frac{1}{2}B\bar{m}(t)Q_2(t),$ where $\bar{m}(t) = m_0 e^{-\varepsilon t} + (aA + 1)B(1 - e^{-\varepsilon t})/\varepsilon.$

Due to network externalities, prices for both products in the closed source model and the price for the complementary product in the open source model begin low in order to quickly establish the network’s installed base and gradually increase as the network size increases, as Figure 1 demonstrates.

We can see that price in the initial periods, for both software and complementary product is zero. This is because of the network externality effect. In order to build the network and extract the maximal surplus, the producer is willing to sacrifice short-term profits for larger future profits.

The behavior of $P_2(t)$ over time depends on the parameters, and there are several of them. The behavior of $P_2(t)$ is difficult to establish analytically, other than the bounds we show, which are increasing over time. Therefore, we conduct a numerical study of the behavior of $P_2(t)$ over time. For this, we fix $\varepsilon = 0.01$ and $\rho = 0.1,$ and examine the behavior of $P_2(t)$ with respect to the remaining parameters.

In the closed source model, hiring is massive early on due to the need to rapidly increase software quality in order to build up the network. As a result, we see software quality increases rapidly in the initial time period. Hiring eases up over time and software quality begins decreasing towards the end of the life of the products. This is quite intuitive since given an anticipated product termination and a zero salvage value of software quality at time $T,$ it is
not profitable to hire people and improve the quality at the end of the life of the product. In the open source model, software quality increases continuously since in this scenario software quality increases as the size of network of users increases, as Figure 2 demonstrates.

<Insert Figure 2 here>

An important difference to note between the open source and closed source is that in the closed source case software quality decreases towards the end of the life of the software, whereas in the open source case it continues to increase. This finding holds for all parameter values. This is because the firm will not find it optimal to invest in quality when the product nears its death, whereas the open source community, not motivated by profit, will continue to improve. This presents interesting welfare implications to open source development, particularly if the benefit to society from increased quality is substantial.

Next, we illustrate the idea that when a firm has to choose between open source and closed source, that choice may depend on the values of the parameters $\alpha$, $k$, $w$, and the initial quality of the software $Q_0$. Suppose that the profits for open source and closed source intersect for some value of $\alpha$. According to Proposition 2, part (a), open source profit is monotonically increasing in $\alpha$ and closed source profit is unchanging in $\alpha$. Therefore, there will be a single point of intersection—a threshold value $\bar{\alpha}$. The numerical analysis clearly confirms that such threshold exists: there appears to be a threshold level of open source community involvement, $\bar{\alpha}$, above which open source will be preferred to closed source (See Figure 3). The parameter $\alpha$ is of particular significance to open source developers, since without substantial community development, the software will fail to take off from its initial stage.

<Insert Figure 3 here>

In-house programmer productivity is the other side of the productivity coin. Suppose that the profits for open source and closed source intersect for some value of $k$. According to Proposition 8, part (a), closed source profit is monotonically increasing in $k$ and open source profit is unchanging in $k$. Therefore, there will be a single point of intersection—a threshold in-house programmer productivity, $\bar{k}$, above which closed source will be preferred to open source (See Figure 4).
In considering whether to develop in-house software, the firm must consider wages as well as in-house productivity. Suppose that the profits for open source and closed source intersect for some value of $w$. According to Proposition 8, part (c), closed source profit is monotonically decreasing in $w$ and open source profit is unchanging in $w$. Therefore, there will be a single point of intersection—a threshold wage factor, $\bar{w}$ - above which open source will be preferred to closed source (See Figure 5).

Finally, initial software quality is a critical consideration. According to Proposition 2, part (b), and According to Proposition 8, part (b), the optimal profit for both open and closed source models are increases with the initial software quality $Q^0_{1}$. When initial quality of the software is high, the firm has little to benefit from opening the source code to the open source community. As such, there exists a threshold initial quality, $Q^0_{1}$, below which open source will be preferred to closed source (See Figure 6). The initial quality of the software appears to have a far greater impact on the closed source profits relative to the open source profits. This is because the software is not sold in the open source case and so the demand for the software only affects profits indirectly through its effect on the demand for the complementary good.

Moreover, we find that in the closed source case, the higher the initial quality of the complementary product, $Q^0_{2}$, the longer the price of the software $P_1$ will remain at zero initially. In addition, the higher the $Q^0_{2}$, the higher the $P_2$ and the lower the $P_1$ at the end of the time horizon.

From figure 7, we see that if the quality of the complementary product is high, the firm can charge more for this complementary product and let the software product be free for a longer duration.
5 Conclusions

Whereas the speed and creativity associated with open source development are well accepted, the profit potential is not always recognized. As companies become increasingly involved in open source projects, the profit implications of open source for companies must be better understood. In this article we argue that the profitability of open source software may come from a product which is a complement to the open source code.

A firm considering open source development as an alternative to closed source development would need to carefully review the relationship between the software in question and the firm’s other products. If the software is found to enhance the usefulness or quality of complementary products and/or if the users of the software and the users of the complementary product belong to the same network, the complementarity can be exploited. By dynamically and simultaneously managing price, product quality, network size, and hiring for both products, the firm will be able to best exploit the complementarity. This effort may or may not benefit from opening the source of the product. Without a clear model of how to exploit freeware, open source development as a substitute to in-house closed source development may be detrimental to firm’s profitability. Our analytical derivation and simulations demonstrate that under various conditions open source may not be beneficial to a firm. On the other hand, there are scenarios where the opposite is the case.

Specifically, open source community involvement is critical to the success of open source initiative. Only above a critical level of community involvement, open source becomes a viable alternative to closed source. To the extent that community involvement can be influenced by the firm’s efforts or influence, such efforts must complement the pricing decisions evaluated here. In-house programmer productivity was shown to be the opposite side of the same coin. More productive and efficient in-house programmers result in less reliance on open source. However, wage is critical in that respect. If programmer productivity is high but wage is higher than some threshold, our results show that open source would be preferred. Finally, if the initial quality of the software is high, development becomes a less critical consideration and extraction of surplus can begin immediately. In such a case, the firm would prefer to charge a positive price for the software and close the code.

We also characterized price, quality, and hiring paths for firms under both the open
source and closed source models. We find that due to network externalities, prices for both products in the closed source model and for the complementary product in the open source model will begin low in order to quickly establish the network’s installed base and gradually increase as the network size increases. We find that in the closed source case, the higher the initial quality of the complementary good, the longer the price of the software will remain at zero initially. That is, when more surplus can be extracted from the complementary good, it may be optimal to have the software as a freeware for a long period of time, even when the source is closed.

In the closed source model, quality will rapidly increase early on to build the network size and begin decreasing towards the end of the life of the product. Similarly, hiring will be massive early on and will ease up over time. This is due to the need to rapidly increase quality in order to establish the network. This is not the case in the open source models. As such, it may be argued that for finitely lived products, open source improves society’s welfare in terms of both quality and productivity.

Future research should examine other forms of revenue extraction from open source as well as the problem from a social planner’s perspective. We would further like to examine the effect of competitive pressures on the firm’s decision to use open source. When the firm opens it source, it may implicitly provide advantages to competitors. Compatibility and legal protection will become paramount considerations when competition is considered.

References


Appendix

PROOF to PROPOSITION 1.

Let \( \tilde{m}(t) = -\varepsilon \tilde{m}(t) \), \( \tilde{m}(0) = m^0 \geq 0 \). Then \( \tilde{m}(t) = m^0 e^{-\varepsilon t} \geq 0 \), for \( 0 \leq t \leq T \). Since 
\[ ah(P_2, m, Q_2)g(0, Q_1) + h(P_2, m, Q_2) \geq 0, \]
it is obvious that \( m(t) \geq \tilde{m}(t) \geq 0 \), where \( m(t) \) is a solution of (6) for arbitrary \( P_2(t) \) and \( Q_1(t) \). Since \( m(t) \geq 0 \), we can similarly conclude that \( Q_1(t) \geq 0 \) for \( 0 \leq t \leq T \). Moreover, it is easy to see that \( m^0 > 0 \) and \( Q^0_1 \) imply \( m(t) > 0 \) and \( Q_1(t) > 0 \) for \( 0 \leq t \leq T \). □

PROOF to PROPOSITION 2.

(a) From (4), we see that \( V^\alpha_2(0, Q_1, m^0) = J(P^*_2(t \mid \alpha)) \) or simply \( V^\alpha_2 = J(P^*_2(t \mid \alpha)) \), where 
\( P^*_2(t \mid \alpha) \) is the optimal price trajectory given \( \alpha \). Let \( Q^{P_2(t) \mid \alpha}_1 \) denote the software quality trajectory given a price trajectory \( P_2(t) \) and \( \alpha \). Similarly, let \( m^{P_2(t) \mid \alpha} \) denote the user network size trajectory given a price trajectory \( P_2(t) \) and \( \alpha \).
From (5), we have \( Q_1(t) = Q_1^0 e^{-\delta t} + \alpha e^{-\delta t} \int_0^t e^{\delta \tau} m(\tau) d\tau \), which increases with \( \alpha \) for every fixed trajectory \( m(t) \). Next we see from (6) that for a given price trajectory \( P_2(t) \geq 0 \),
\[
\frac{\partial \bar{m}(t)}{\partial Q_1(t)} = ah(P_2(t), m(t), Q_2(t)) \frac{\partial g(t)}{\partial Q_1(t)} \geq 0,
\]
which means that \( \bar{m}(t) \) increases with \( Q_1(t) \). This implies that both \( Q_1(t) \) and \( m(t) \) increase as \( \alpha \) increases for a given price trajectory \( P_2(t) \geq 0 \).

Let \( 0 < \alpha_1 < \alpha_2 \). Then \( Q_1^{P_2(t)}(t \mid \alpha_1) \leq Q_1^{P_2(t)}(t \mid \alpha_2) \) and \( m^{P_2(t)}(t \mid \alpha_1) \leq m^{P_2(t)}(t \mid \alpha_2) \). By the assumption on the functions \( h \) and \( \sigma \), it is apparent that \( J(P_2(t \mid \alpha_1) \leq J(P_2(t \mid \alpha_2) \). By definition, \( J(P_2(t \mid \alpha_1) \leq J(P_2^+(t \mid \alpha_1) \leq J(P_2(t \mid \alpha_2) \leq J(P_2^+(t \mid \alpha_2) \).

Therefore, \( V_O^{\alpha_1} \leq V_O^{\alpha_2} \). This completes the proof.

(b) The proof is similar to part (a).

PROOF to PROPOSITION 3.

From Proposition 2, part (b), we know that
\[
\frac{\partial V_O(0, Q_1(0))}{\partial Q_1(0)} \geq 0.
\]
Therefore, \( \lambda(0) = \frac{\partial V_O(0, Q_1(0))}{\partial Q_1(0)} \geq 0 \). The same argument extends to \( \lambda(t) = \frac{\partial V_O(0, Q_1(t))}{\partial Q_1(t)} \geq 0 \).

PROOF to PROPOSITION 4. This proof requires Lemma 1.

**LEMMA 1.** In the open source model, \( P_2 + \mu(\alpha_0(0, Q_1) + 1) \geq 0 \), for \( 0 \leq t \leq T \).

PROOF to LEMMA 1.

According to (11) and (12), we know that there are two cases:

\textbf{Case1:}
\[
\left\{ h(P_2, m, Q_2) + \left[ P_2 + \mu(\alpha(0, Q_1) + 1) \right] \frac{\partial h}{\partial P_2} \right\} \bigg|_{P_2=0} \leq 0 \text{ and } \eta_2 \geq 0,
\]
and

\textbf{Case2:}
\[
\left\{ h(P_2, m, Q_2) + \left[ P_2 + \mu(\alpha(0, Q_1) + 1) \right] \frac{\partial h}{\partial P_2} \right\} \bigg|_{P_2=0} = 0 \text{ and } \eta_2 = 0.
\]

In case 1, \( [P_2 + \mu(\alpha(0, Q_1) + 1)]|_{P_2=0} = \mu(\alpha(0, Q_1) + 1) \geq -h(P_2, m, Q_2)/\frac{\partial h}{\partial P_2}|_{P_2=0} \geq 0 \).

In case 2, \( [P_2 + \mu(\alpha(0, Q_1) + 1)]|_{P_2=0} = -h(P_2, m, Q_2)/\frac{\partial h}{\partial P_2}|_{P_2=0} \geq 0 \). The result follows.

By contradiction. Suppose at an arbitrarily chosen time \( \tau \in [0, T] \), \( \mu(\tau) < 0 \). By Proposition 3 and Lemma 1,
\[
\dot{\mu} = (\rho + \varepsilon)\mu - \alpha \lambda - [P_2 + \mu(\alpha(0, Q_1) + 1)] \frac{\partial h}{\partial m} < 0.
\]
Therefore, $\mu(\tau) < 0$ for $\tau \leq t \leq T$. This contradicts $\mu(T) \geq 0$. So $\mu(\tau) \geq 0$. Since $\tau$ is arbitrary, we can conclude that $\mu(T) \geq 0$ for $0 \leq t \leq T$. □

**Proof to Proposition 5.**

The salvage value software quality is zero at time $T$. Therefore, $\lambda(T) = 0$. By assumption, $h \geq 0$ and $\frac{\partial g}{\partial Q_1}$. From these assumptions and Proposition 4,

$$a\mu h(P_2, m, Q_2) \frac{\partial g}{\partial Q_1} \geq 0.$$ 

Let $f(t) = a\mu h(P_2, m, Q_2) \frac{\partial g}{\partial Q_1}$. Equation (9) becomes

$$\dot{\lambda} = (\rho + \delta)\lambda - f(t), \lambda(T) = 0.$$

Therefore,

$$\lambda(t) = e^{(\rho + \delta)t} \int_t^T e^{-(\rho + \delta)\tau} f(\tau) d\tau.$$

Since $f(t) \geq 0$, we conclude that $\lambda(t)$ decreases over time. □

**Proof to Proposition 6.**

If $P_2^* > 0$, then

$$h + P_2 \frac{\partial h}{\partial P_2} + \mu(\rho g(0, Q_1) + 1) \frac{\partial h}{\partial P_2} \bigg|_{P_2^*} = 0 \text{ (from (11)).}$$

By Proposition 4 and the assumptions that $g \geq 0$ and $\frac{\partial h}{\partial P_2} \leq 0$, we have $\mu(\rho g(0, Q_1) + 1) \frac{\partial h}{\partial P_2} \bigg|_{P_2^*} \leq 0$.

Therefore,$h + P_2 \frac{\partial h}{\partial P_2} \bigg|_{P_2^*} \geq 0$. By definition, $rac{\partial F_O}{\partial P_2} \bigg|_{P_2^*} = h + P_2 \frac{\partial h}{\partial P_2} \bigg|_{P_2^*} = 0$. Then $P_2 = h + P_2 \frac{\partial h}{\partial P_2} \bigg|_{P_2^*} \geq 0$. By the concavity of $F_O$, we conclude $P_2^*(t) \leq \hat{P}_2(m^*(t), Q_2(t))$ for $0 \leq t \leq T$. Moreover, if the salvage value is zero at time $T$, then $\mu(T) = 0$. From the previous argument, it is easy to show $P_2^*(T) \leq \hat{P}_2(m^*(T), Q_2(T))$. □

**Proof to Proposition 7.** The proof is similar to that of Proposition 1. □

**Proof to Proposition 8.**

The proofs for part (a) and (b) are similar to that of Proposition 2. (c) Using the Envelope Theorem (e.g., Varian, 1978, Page 268), we have

$$\frac{dV_C}{dw} = \frac{\partial L}{\partial w} = -\int_0^T N^2 dt.$$ 

Therefore, the optimal closed source profit decreases with $w$. □

**Proof to Proposition 9.** The proof is similar to that of Proposition 3. □
**Proof to Proposition 10.**

The proof requires Lemma 2 and Proposition 9.

**Lemma 2.** In the closed source model, \( P_1 + a\mu \geq 0 \) and \( P_2 + \mu + (P_1 + a\mu)g(P_1, Q_1) \geq 0 \), for \( 0 \leq t \leq T \).

**Proof to Lemma 2.** The proof is similar to that of Lemma 1. □

By contradiction. Suppose at an arbitrarily chosen time \( \tau \in [0, T] \), \( \mu(\tau) < 0 \). By Proposition 9 and Lemma 2,

\[
\dot{\mu} = (\rho + \epsilon)\mu - [P_2 + \mu + (P_1 + a\mu)g(P_1, Q_1)] \frac{\partial h}{\partial m} < 0.
\]

Therefore, \( \mu(\tau) < 0 \) for \( \tau \leq t \leq T \). This contradicts \( \mu(T) \geq 0 \). So \( \mu(\tau) \geq 0 \). Since \( \tau \) is arbitrary, we can conclude that \( \mu(T) \geq 0 \) for \( 0 \leq t \leq T \). □

**Proof to Proposition 11.**

The proof of \( \lambda \) decreasing is along the lines of Proposition 5. Following (22) and (25), we have

\[
N = \begin{cases} 
  k\lambda/(2w), & k\lambda/(2w) \geq 0 \\
  0, & k\lambda/(2w) < 0 
\end{cases}.
\]

Since \( \lambda(t) \geq 0 \) for \( 0 \leq t \leq T \), it must be that \( N = k\lambda/(2w) \) for \( 0 \leq t \leq T \). Therefore, the trajectory of \( N \) is the same as that of \( \lambda \): it starts positive and declines to zero by time \( T \). □

**Proof to Corollary 1.**

From (14), \( \dot{Q}_1 = kN - \delta Q_1 \). From Proposition 11, we know the number of in-house programmers \( N \) starts positive and declines to zero by time \( T \). Therefore, \( \dot{Q}_1 \) is positive at the very beginning. Also \( kN \) declines over time and \( \delta Q_1 \) increases over time as long as \( \dot{Q}_1 > 0 \). Therefore, there exists a time \( \tau \) such that \( \dot{Q}_1(\tau) = kN(\tau) - \delta Q_1(\tau) = 0 \). Note that \( \dot{Q}_1(t) = k\dot{N}(t) - \delta \dot{Q}_1(t) = 0 \) for \( t \geq \tau \) with \( \dot{Q}_1(\tau) = 0 \). This is a first order equation for \( \dot{Q}_1(t) \), whose solution is \( \dot{Q}_1(t) = ke^{-\delta t} \int_\tau^t e^{\delta s} \dot{N}(s)ds \leq 0 \), since \( \dot{N}(s) \leq 0 \). Thus \( Q_1(t) \) decreases for \( t \geq \tau \). □

**Proof to Proposition 12.**
If \( P_1^* > 0 \), then \( \left( g(P_1, Q_1) + P_1 \frac{\partial g}{\partial P_1} \right) h(P_2, m, Q_2) + a\mu \frac{\partial g}{\partial P_1} h(P_2, m, Q_2) \bigg|_{P_1^*, P_2^*} = 0 \) (from (20)). We can also say that \( g(P_1, Q_1) + P_1 \frac{\partial g}{\partial P_1} + a\mu \frac{\partial g}{\partial P_1} \bigg|_{P_1^*} = 0 \). By Proposition 10 and the assumption that \( \frac{\partial g}{\partial P_1} \leq 0 \), we have \( a\mu \frac{\partial g}{\partial P_1} \bigg|_{P_1^*} \leq 0 \). Therefore, \( g(P_1, Q_1) + P_1 \frac{\partial g}{\partial P_1} \bigg|_{P_1^*} \geq 0 \). By definition, \( \frac{\partial F_C}{\partial P_1} \bigg|_{P_1^*, P_2^*} = \left( g(P_1, Q_1) + P_1 \frac{\partial g}{\partial P_1} \right) h(P_2, m, Q_2) \bigg|_{P_1^*, P_2^*} = 0 \). We can also say that \( g(P_1, Q_1) + P_1 \frac{\partial g}{\partial P_1} \bigg|_{P_1^*} = 0 \). Then \( \hat{P}_1 = -g/\partial g/\partial P_1 \bigg|_{\hat{P}_1} \geq 0 \). By the concavity of \( F_C \), we conclude \( P_1^*(t) \leq \hat{P}_1(m^*(t), Q_1(t), Q_2(t)) \) for \( 0 \leq t \leq T \). Similarly, if \( P_2^* > 0 \), then \( h(P_2, m, Q_2) + [P_2 + P_1 g(P_1, Q_1)] \frac{\partial h}{\partial P_2} + \mu (ag(P_1, Q_1) + 1) \frac{\partial h}{\partial P_2} \bigg|_{P_1^*, P_2^*} = 0 \) (from (21)). Clearly, \( P_2^* \) is a function of \( P_1^* \). Let \( P_2^*(P_1) \) is the solution to \( h(P_2, m, Q_2) + [P_2 + P_1 g(P_1, Q_1)] \frac{\partial h}{\partial P_2} + \mu (ag(P_1, Q_1) + 1) \frac{\partial h}{\partial P_2} \bigg|_{P_1^*, P_2^*} = 0 \). It can be shown that \( P_2^*(P_1^*) \leq P_2^*(\hat{P}_1) \). By Proposition 10 and the assumptions that \( g \geq 0 \) and \( \frac{\partial h}{\partial P_2} \leq 0 \), we have \( \mu (ag(P_1, Q_1) + 1) \frac{\partial h}{\partial P_2} \bigg|_{P_1^*, P_2^*} \leq 0 \). Therefore, \( h(P_2, m, Q_2) + [P_2 + P_1 g(P_1, Q_1)] \frac{\partial h}{\partial P_2} \bigg|_{P_1^*, P_2^*} \geq 0 \). By definition, \( \frac{\partial F_C}{\partial P_2} \bigg|_{P_1^*, P_2^*} = h(P_2, m, Q_2) + [P_2 + P_1 g(P_1, Q_1)] \frac{\partial h}{\partial P_2} \bigg|_{P_1^*, P_2^*} = 0 \). Clearly, \( \hat{P}_2 \) is a function of \( \hat{P}_1 \). We denote it as \( \hat{P}_2(\hat{P}_1) \).

By the concavity of \( F_C \), we know that \( P_2^*(\hat{P}_1) \leq \hat{P}_2(\hat{P}_1) \). Therefore, \( P_2^*(P_1^*) \leq \hat{P}_2(\hat{P}_1) \). we conclude \( P_2^*(t) \leq \hat{P}_2(m^*(t), Q_1(t), Q_2(t)) \) for \( 0 \leq t \leq T \). Moreover, if the salvage value is zero at time \( T \), then \( \mu(T) = 0 \). From the previous argument, it is easy to show \( P_1^*(T) \leq \hat{P}_1(m^*(T), Q_1(t), Q_2(T)) \) and \( P_2^*(T) \leq \hat{P}_2(m^*(T), Q_1(t), Q_2(T)) \).

\[
\text{PROOF to COROLLARY 2.}
\]

(i) Exponential demand function. From Proposition 13, \( P_2^*(t) \leq m^*(t)Q_2(t) \). From (6), \( \hat{m} = (a \exp(-\frac{c}{Q_1}) + 1) \exp(-\frac{P_2}{mQ_2}) - \varepsilon m, m(0) = m^0 \). Let \( \hat{m} = (a + 1) - \varepsilon m, \hat{m}(0) = m^0 \). Then \( \hat{m}(t) = m^0 e^{-\varepsilon t} + (a + 1)(1 - e^{-\varepsilon t})/\varepsilon, 0 \leq t \leq T \). Clearly, \( \hat{m} > \hat{m} \). The result follows.

(ii) Linear-price demand function. Proof is similar to (i).

\[
\text{PROOF to COROLLARY 3.}
\]

(i) Exponential demand function. From Proposition 13, \( P_1^*(t) \leq Q_1^*(t) \). From (14), \( \dot{Q}_1 = kN - \delta Q_1, Q_1(0) = Q_1^0 \). Let \( \dot{Q}_1 = kN(0) - \delta \tilde{Q}_1, \tilde{Q}_1(0) = Q_1^0 \), where \( N(0) \) is the number of in-house programmers at time 0. Then \( \dot{Q}_1(t) = Q_1^0 e^{-\delta t} + kN(0)(1 - e^{-\delta t})/\delta, 0 \leq t \leq T \). \( \dot{Q}_1 > \dot{Q}_1 \) since \( N \) is decreasing over time. The result follows.
From Proposition 13, $P^*_2(t) \leq m^*(t)Q_2(t) - \hat{P}_1^*(t)g(\hat{P}_1^*(t), Q_1^*(t)) \leq m^*(t)Q_2(t)$. From (15), \( \dot{m} = (a \exp(-\frac{P_1}{Q_1}) + 1) \exp(-\frac{P_2}{mQ_2}) - \varepsilon m, m(0) = m^0 \). Let \( \tilde{m} = a + 1 - \varepsilon m, \tilde{m}(0) = m^0 \). Then \( \tilde{m}(t) = m^0 e^{-\varepsilon t} + (a + 1)(1 - e^{-\varepsilon t})/\varepsilon, 0 \leq t \leq T \). Clearly, \( \hat{m} > \tilde{m} \). The result follows.

(ii) Linear-price demand function. Proof is similar to (i). \qed
Figure 1: Prices and their upper bounds over time.

Closed Source

Exponential Demand

Priced Linear Demand

Open Source

Exponential Demand

Priced Linear Demand
Figure 2: Quality and hiring path for finite horizon.

Closed Source

Exponential Demand

Price
- linear Demand

Open Source

Exponential Demand

Profit over different values of $\alpha$ for open source and closed source.
Figure 4: Profit over different values of $k$ for open source and closed source.

Figure 5: Profit over different values of $w$ for open source and closed source.

Figure 6: Profit over levels of initial quality $Q_1^0$. 
Figure 7: The effects of $Q_2^0$ on $P_1$ and $P_2$ for closed source.

$Q_2^0 = 1$

Exponential Demand

$Q_2^0 = 2$

Exponential Demand

$Q_2^0 = 3$

Exponential Demand

Linear Demand

Linear Demand

Linear Demand

Linear Demand

Linear Demand