A Strategic Analysis of Competition Between Open Source and Proprietary Software

Ravi Sen, Assistant Professor
Information and Operations Management Department
320 Wehner Building, Mays Business School,
Texas A&M University, College Station TX-77843-4217
Phone: (979) 845-0659
Fax: (979) 845-5653
Email: rsen@mays.tamu.edu

A Strategic Analysis of Competition between Open Source and Proprietary Software

Abstract: This paper takes an analytical approach to identify the conditions under which freely available open source software (OSS) and/or the commercial version of the same (OSS-SS) will adversely affect the market position of proprietary software (PS), and suggests some strategic steps that the PS vendor can take in order to compete successfully.

For example, we find that in software markets characterized by low network benefits and OSS-SS with low usability (relative to PS), open source software will have the dominant market share. Interestingly, in these markets the profitability of PS vendor, when the OSS-SS is also present in the market, is higher than its profitability, when the OSS-SS is absent from the software market.

In software markets characterized by low network benefits and OSS-SS with high usability, PS will dominate the market in terms of market share. It can maintain its domination by actively participating in OSS projects and ensuring that OSS is as usable as OSS-SS.

In software markets characterized by high network benefits and OSS-SS with low usability (relative to PS), we should expect to see the open source software dominating this market in future. However, PS vendors can effectively compete by ensuring that PS is more usable than OSS-SS and OSS.

Finally, in software markets characterized by high network benefits and OSS-SS with high usability, PS faces the maximum threat since open source software will dominate the market in terms of market share. Furthermore, the equilibrium price that the PS can

charge will not result in positive profits, thus ensuring the exit of PS vendors from the software markets. However, we have yet to see a commercial version of open source in this software category that is as usable as the PS. Therefore, we have not observed the exit of PS vendors from this software segment.

Keywords: Open source software, software market, software competition, economics of open source, commercial open source, FLOSS.

A Strategic Analysis of Competition between Open Source and Proprietary Software

1. INTRODUCTION

As opposed to proprietary commercial software, open source software (OSS) allows the users to have access to the source code of the software, the freedom to use the software as they seem fit, improve it, fix its bugs, augment its functionality, and redistribute the software (for free, or at a charge) to other users, who could themselves modify and/or use it according to their own needs. Apache (a web server), Linux (an operating system), and Sendmail (an Internet mail transfer agent) are some notable examples of open source software that have achieved remarkable success both in technical and economic terms. Some other examples of proprietary software and their open source competition is given in *Table 1*,

Table 1: Open Source and Proprietary Applications		
Application	**Proprietary**	**Open Source**
Web Browser	Internet Explorer	Mozilla Firefox
Web Server	Microsoft IIS, Netscape	Apache
Application Server	WebMethods	JBoss, Apache Tomcat
Office Suite	MS Office, Corel WordPerfect Office	OpenOffice.org (based on Sun's proprietary StarOffice)
E-mail/ Collaboration	MS Exchange, Lotus Notes	Ximian Evolution
Database	Oracle9i, DB2, MS SQL	MySQL, PostgreSQL

The success of these open source software poses some concern for proprietary (or "closed source") software vendors. This concern is best summed up in the following statement from a memo written by Microsoft CEO Steve Ballmer in 2003[1]- "…*Noncommercial software products in general and Linux in particular present a competitive challenge for us and our entire industry and they require our concentrated focus and attention.*…."

[1] http://www.topdog04.com/ooo216.html

This statement illustrates the seriousness with which the commercial software producers are treating competition from open source software, and their concern is not misplaced. OSS user-license characteristics combined with the fact that they available for free, is resulting in their increasing adoption by the user population. In a November 2003 CIO survey of 375 information executives, 54% said that within five years open source would be their dominant platform (Koch 2003). Big vendors (e.g. Dell, IBM, and Sun) have lined up to support it or port their applications to it (e.g. SAP to Linux platforms), and major enterprises are running mission-critical functions on open-source IT.

These commercial successes of open source software have led to an increasing interest in understanding this form of software, its development process, and its impact on software industry. The current research in this area can be classified into three broad categories- (a) understanding the motivation on the part of individuals and organizations to initiate and participate in open source projects; (2) adoption of open source applications by individuals and organizations; and (3) understanding the impact of open source software applications on the industrial organization of software industry and social welfare. Most of the current literature on open source projects focuses on the motivation of individuals and organizations to initiate and participate in these projects (e.g. Green; Moon and Sproull 2000; Raymond 2000; Learner and Tirole 2002; Johnson 2002; Mustonen 2003). Furthermore, adoption of open source applications by organizations could be explained by the existing theories on software diffusion and adoption, e.g. Technology Acceptance Model (Davis 1989) and its numerous extensions (e.g. Hartwick and Barki 1994; Karhanna and Straub 1999; Venkatesh and Davis, 2000; Moore and Benbesat 1991; Adam *et al.* 1992). Some work has also been done to study software markets from public

policy point of view (e.g., Katz and Shapiro 1999), and the type of software license best for the society (e.g., Gaudeul 2004). There are a few studies that analyze the competition between open source and commercial software. However, these papers consider open source to be privately- provided public goods and model the choice between producing open source software and purchasing their commercial substitute. This choice is modeled from user-developers' point of view (e.g. Johnson 2002; Bessen 2002). User-developers are individual/organizations that not only participate as developers in open source projects but are also the primary users of the end product. The user-developers, however, form a very small proportion of most software markets, which are generally dominated by users who are never involved in the development of the software[2] and it is this market where the main competition takes place. However, there is a dearth of work that can help us understand the impact of OSS on competition in this segment of the market- i.e., the traditional commercial software industry- which is the main concern for commercial software vendors like Microsoft. Furthermore, there is no study which analyses the extent of this competition, i.e. is the competition serious enough to result in the demise of commercial software vendors and by extension the end of software industry as we know it, or will the commercial software vendors survive this competition and co-exist with open source software? Finally, the commercial software vendors would like to know about their strategic options for competing successfully with any emerging threat of open source software.

This paper takes a purely analytical approach to understand the competition between open source and proprietary software. It differs from existing papers of similar nature

[2] Some of these end users could be involved as beta testers, but they also do not contribute any code to the software.

because it presents a mathematical model that not only captures the competition between commercial software and open source software, but also incorporates user specific characteristics such as their valuation of software usability, and software specific characteristics such as its usability and the strength of network benefits provided by the software. This model is then analyzed to explain the impact of open source software on traditional commercial software industry and identify the conditions under which it can have an adverse impact on the existence of commercial software vendors. The paper is organized as follows. *Section 2* defines the model that represents competition between commercial software and open source software, *Section 3* solves the model for equilibrium outcomes, and *Section 4* presents the results and their implications for the software industry followed by the conclusions.

2. THE MODEL

The proposed model does not dwell upon the reasons for the existence of the open source substitutes. These exist because there are individuals and organizations willing to develop and distribute them. The existing literature does a good job of explaining the motivation of these individuals and organizations to initiate, develop and distribute open source software (e.g. Moon and Sproull 2000; Raymond 2000; Learner and Tirole 2002; Johnson 2002; Mustonen 2003). Instead we assume that both types of software exist and compete in their relevant market segment. It should be noted at this point that the competition is assumed to be between the same categories of software, i.e. the competing software should be targeted at the same user needs. The objective of this paper is- (a) to analyze this competition and determine the conditions under which OSS can have an adverse impact on the existence of PS vendors in the same software category, and (b) suggest

strategic solutions to PS vendors, which can help them compete successfully against open source software.

2.1 Software Characteristics: The model assumes that the software under consideration, whether open source or proprietary meet the functional (i.e. whether it can perform some function) and reliability (e.g., error handling, security feature)[3] needs of the potential users. However, they differ in their usability. Overall software usability is typically described in terms of five characteristics- ease of learning, efficiency of use, memorability, error frequency and severity, and subjective satisfaction (Nielsen 1993). In this model, we define the lack of software usability (i.e. *un-usability*) for any particular software, by β. Proprietary software (PS) is assumed to have the highest software usability, i.e. $\beta = 0$ because of features like extensive manuals that come with these software; multi-channel support provided by the software vendor; and efforts by the vendor to make the software as user friendly as possible (e.g. default installation options, intuitive GUI, contextual help, demos and examples, and fault-tolerance features of the software etc). On the other hand, OSS without support services have the lowest average software usability (Nichols and Twidale 2003; Frishberg *et al.* 2002), and are assumed have $\beta = 1$. Commercial OSS (denoted by OSS-SS) that comes bundled with support services (e.g. documentation, phone consultation, training etc.) is assumed to have a $0 < \beta \leq 1$. Thus, for a vendor of OSS-SS, the level of its usability is a decision variable that it can influence in order to maximize its profits. However, it has to balance the desire for high usability among users with the cost of providing highly usable software. Finally, the proposed model assumes that users benefit from positive network effects generated by

[3] This paper assumes both PS and OSS to be equally reliable because there is a lack of consensus about which form of software is more secure- PS or OSS (e.g. Viega 2004).

software's expected installation base. The assumption is based on the existing literature rooted in industrial organizational theories that have established the positive network externalities of software (e.g. Farrel and Saloner 1985, 1986; Katz and Shapiro 1986). The total network affect for any software is θq_e, where q_e is the expected installation base of the software and θ $(0 \le \theta \le 1)$ is the strength of network affect. Software which do not benefit much from network affect (direct or indirect) have low value for θ, while software that depend on network benefits for their success have a relatively high value for θ.

2.2 Software Market: The basic setup of the model involves a software market, which consists of *commercial proprietary software* (PS), an *open source substitute with documentation support* (OSS-SS), and this same *open source substitute without documentation support* (OSS).

- **Packaged Software (PS)**: This includes commercial software that come with a full documentation and support services. The total cost of ownership for this software is P_p which is assumed to be the sum of the purchase price and the present discounted value of the per-period service/support cost. The model assumes $(1+\beta)c$ to be the marginal cost associated with each unit of software sold (e.g. cost of after sales support service). Thus the marginal cost is higher for software that has less usability since users of such software would require more help with using the software. However, since PS is assumed to be highly usable, (i.e. $\beta = 0$), its marginal cost is just c, and the PS vendor's profit function is $\pi_P = [P_P - c]D_P$, where D_P is the demand for PS.

- **Open Source Software with Support Services (OSS-SS)**: This includes a *non-divisible bundle* of *highly usable open source software and accompanying support services for this software*. The total cost of ownership for this software bundle is denoted by P_S. The marginal cost of servicing the client base is assumed to be $(1+\beta)c$, and the profit function for OSS-SS vendor is $\pi_S = [P_S - (1+\beta)c]D_S$, where the demand for OSS-SS is D_S, and β denotes the lack of usability in OSS-SS.
- **Open Source Software without Support Services (OSS)**: This includes open source software (both in executable form and its actual code) that can be downloaded for free from the Internet. A user can seek (e.g. on the Internet) additional help on using this software, but the "free" help that is available is not the same quality as the one provided by OSS-support-services vendors, and therefore this help does not have any significant impact on the usability of the software. The demand for this type of software is denoted by D_O in the model.

2.2 Software Users: Software users are indexed by their valuation of software usability, $0 < v \le V$,[4] and are distributed uniformly over this interval with unit density. This valuation can be interpreted as the value of time spent in installing, configuring, and maintaining the software for proper use. Consumer of type *v* has the utility-

$$U(v) = A + \theta q_e - (1+\beta_i)v - P_i \qquad \textbf{(1)}$$

Where $i \in (PS, OSS-SS, OSS)$; A is the inherent value of the software for the user; and q_e is the expected market share of the software. A user opts for the software that offers the maximum utility, i.e. $q_e - P_i - (1+\beta_i)v$.

[4] For instance, highly skilled and experienced software users might not place as much emphasis on software usability as those users who are relatively less skilled or experienced in the use of software.

The variables, parameters used in the model and the functional relationships between these variables are summarized in *Table 2*.

Table 2: Assumptions about software and user characteristics

	Proprietary Software (PS)	Commercial Open Source (OSS-SS)	Free Open Source (OSS)
Inherent Value of the SW	A	A	A
Lack of S/W Usability	$\beta_{PS} = 0$	$0 < \beta_{OSS-SS} = \beta \leq 1$	$\beta_{OSS} = 1$
Strength of NW Affect	$0 \leq \theta \leq 1$	$0 \leq \theta \leq 1$	$0 \leq \theta \leq 1$
Expected Installation Base	$q_P = q$	$1 - q_P = 1 - q$	$1 - q_P = 1 - q$
Total NW Affect	θq	$\theta(1-q)$ *	$\theta(1-q)$ *
Consumers valuation of SW (un)usability	$0 < v \leq V$	$0 < v \leq V$	$0 < v \leq V$
Price Paid for the Software	P_P	P_S	0
Demand	D_P	D_S	D_O
Profit Functions	$\pi_P = [P_P - c]D_P$	$\pi_P = [P_S - (1+\beta)c]D_S$	0
Utility Function for User of Type *v*	$U_P(v) = A + \theta q - v - P_P$	$U_S(v) = A + \theta(1-q) - (1+\beta)v - P_S$	$U_O(v) = A + \theta(1-q) - 2v$

* Users of open source software benefit from overall installation base of this type of software, irrespective of whether the users install the "commercial" open source version or the "free" open source version.

3. EQUILIBRIUM IN THE PRICE-SETTING GAME

The price equilibrium is determined for a simple scenario consisting of proprietary software (i.e. PS), commercial open source software bundle (i.e. OSS-SS), and free open source software (i.e. OSS). Since fixed costs are irrelevant to the pricing game, they are assumed to be zero. The equilibrium prices, demand, and profits are derived in *Appendix A*. The equilibrium prices and demand are as follows:

$$P_P^* = \left(\frac{2(1-2\theta)(\beta V - \theta) + (1-2\theta)(3+\beta)c + \theta(1-\beta)(1-2V)}{3+\beta-8\theta} \right) \quad \textbf{(2a)}$$

$$P_S^* = \left(\frac{2\theta(1-\beta)(1-2V) + (1-\beta)(\beta V - \theta) + [3+\beta+4\theta(1+\beta)]c}{3+\beta-8\theta} \right) \quad \textbf{(2b)}$$

$$D_{PS}^* = \left[\left(\frac{\beta V - \theta}{\beta - 2\theta} \right) - \left(\frac{(1+\beta-4\theta)(\beta V - \theta) - \theta(1-\beta)(1-2V) - 2\theta(5+3\beta)c}{(\beta-2\theta)(3+\beta-8\theta)} \right) \right] \quad \textbf{(3a)}$$

$$D_{OSS-SS}^* = \left[\begin{array}{l} \left(\frac{\theta(1-2V)}{(\beta-2\theta)} + \frac{(1+\beta-4\theta)(\beta V - \theta) - \theta(1-\beta)(1-2V) - 2\theta(5+3\beta)c}{(\beta-2\theta)(3+\beta-8\theta)} \right) - \\ \left(\frac{2\theta(1-2V) + (\beta V - \theta)}{3+\beta-8\theta} \right) - \left(\frac{3+\beta+4\theta(1+\beta)}{(3+\beta-8\theta)(1-\beta)} \right) c \end{array} \right] \quad \textbf{(3b)}$$

$$D_{OSS}^* = \left(\frac{2\theta(1-2V) + (\beta V - \theta)}{3+\beta-8\theta} \right) + \left(\frac{3+\beta+4\theta(1+\beta)}{(3+\beta-8\theta)(1-\beta)} \right) c \quad \textbf{(3c)}$$

In order to simplify the analysis, we will look at scenarios representing two broad categories of software. These software categories are differentiated by the strength of network affects (θ), i.e. for $\theta = 0$ (no network affects present) and $\theta = 1$ (maximum network affects present). For instance, when there are standards which allow multiple software (in the same category) to interoperate, the network affects the entire user population of these software, resulting in low value of θ for individual software users. This is true for software such as server operating systems (e.g. Linux, Windows NT), and web servers (e.g. Apache, MS-IIS). In these software markets there are extremely strong market pressures to interoperate with pre-existing or most widely accepted standards. Another instance when network affect is not strong is when the software is a stand-alone application (e.g. CD writer, anti-virus, personal firewall). The major part of the value derived from the use of these software tools is independent of the number of other users of these software applications. Although, there are some indirect network affects present

(e.g. a large installation base could result in better support network), these are not strong enough. On the other hand, in software markets where the competing products are based on proprietary standards and protocols, we are likely to see strong network affects playing a significant role in defining the competition. A good example is that of office productivity applications where MS-Office users benefit from its large installation base. Another example is that of desktop operating systems, where users benefit from strong indirect network affects (e.g. more applications are available for the operating system with high market share). Table 3 illustrates this classification of software markets by providing relevant examples in each category.

Table 3: Software Market Classification Based on Network Affects

Example: Strength of NW Affects Low	Example: Strength of NW Affects High
• Desktop stand-alone single user applications (e.g. PC diagnostic tools, single player PC games, Personal firewalls, CD writers, web browsers such as Firefox and Explorer, e-mail client such as Thunderbird and Outlook). • Infrastructure software based on universally accepted standards and protocols (e.g. Web Servers such as Apache and IIS, DNS servers such as BIND, and email servers)	• Desktop office productivity software (e.g. MS Office, OpenOffice) • Database servers (e.g. Oracle, MySQL) • Network operating system (e.g. Windows 2000, RedHat Linux, Novel Netware) • Desktop operating systems (e.g. Windows XP, Suse Linux9)

The equilibrium statistics for the two scenarios are given in *Table 4*.

Table 4: Equilibrium Statistics for Scenario#1 and Scenario #2

Equilibrium Statistics for Scenario#1 (i.e. When $\theta = 0$)

	Demand*	Price*	Profit*
PS Vendor	$D_P^* = \frac{2V}{(3+\beta)}$	$P_P^* = \frac{2\beta V}{3+\beta} + c$	$\pi_P^* = \left[\frac{4\beta V^2}{(3+\beta)^2}\right]$
OSS-SS Vendor	$D_S^* = \frac{V}{(3+\beta)} - \frac{c}{(1-\beta)}$	$P_S^* = \frac{\beta V(1-\beta)}{(3+\beta)} + c$	$\pi_S^* = \left[\frac{V}{(3+\beta)} - \frac{c}{(1-\beta)}\right]\left[\left(\frac{\beta V(1-\beta)}{3+\beta}\right) - \beta c\right]$
OSS	$D_O^* = \frac{\beta V}{(3+\beta)} + \frac{c}{(1-\beta)}$	zero	zero

Equilibrium Statistics for Scenario#2 (i.e. When $\theta = 1$)

	Demand*	Price*	Profit*
PS Vendor	$D_P^* = \left(\frac{3-\beta-2V+2(5+3\beta)c}{(\beta-2)(\beta-5)}\right)$	$P_P^* = \left(\left(\frac{3-\beta-2V}{\beta-5}\right) - \left(\frac{(3+\beta)c}{\beta-5}\right)\right)$	$\pi_P^* = \left(\left(\frac{3-\beta-2V+2(5+3\beta)c}{(\beta-2)(\beta-5)}\right)X \left(\left(\frac{3-\beta-2V}{\beta-5}\right) - \left(\frac{(3+\beta)c}{\beta-5}\right) - c\right)\right)$
OSS-SS Vendor	$D_S^* = \left(\left(\frac{V(4-\beta)-1-2(5+3\beta)c}{(\beta-2)(\beta-5)}\right) - \left(\frac{7+4\beta}{(\beta-5)(1-\beta)}\right)c\right)$	$P_S^* = \left(\left(\frac{(1-\beta)[1-V(4-\beta)]}{\beta-5}\right) + \left(\frac{(7+4\beta)c}{\beta-5}\right)\right)$	$\pi_S^* = \left(\left(\left(\frac{V(4-\beta)-1-2(5+3\beta)c}{(\beta-2)(\beta-5)}\right) - \left(\frac{7+4\beta}{(\beta-5)(1-\beta)}\right)c\right)X \left(\left(\frac{(1-\beta)[1-V(4-\beta)]}{\beta-5}\right) + \left(\frac{(7+4\beta)c}{\beta-5}\right) - (1+\beta)c\right)\right)$
OSS	$D_O^* = \left(\left(\frac{1-V(4-\beta)}{\beta-5}\right) + \left(\frac{7+4\beta}{(\beta-5)(1-\beta)}\right)c\right)$	zero	zero

Analysis of the equilibrium prices (*Table 4*) leads to the following three propositions:

Proposition 1*: PS prices are always greater than OSS-SS prices for software that displays weak network affects (i.e.* $\theta = 0$ *).*

This result is evident in markets for software such as server operating systems (e.g. Linux, Windows NT), and web servers (e.g. Apache, MS-IIS), and is illustrated in *Figure 1* that shows the affect of the level of (un)usability on the equilibrium prices for proprietary software (PS), and commercial open source software (i.e. OSS-SS).

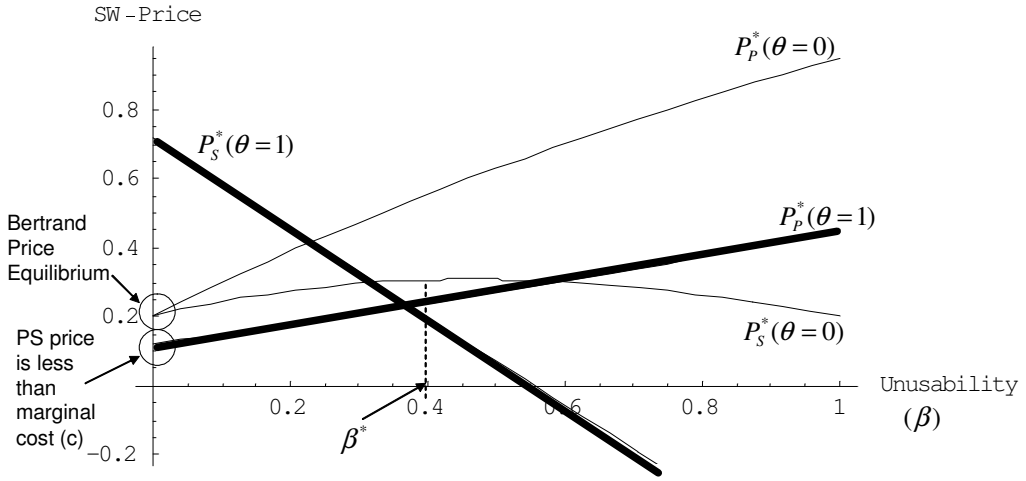

Figure 1: Equilibrium SW Prices $[Assume: V = 1.5; c = 0.2]$

Proposition 2*: When software offers weak network benefits (i.e.* $\theta = 0$*), the equilibrium prices for OSS-SS are maximum at some value of* $0 < \beta = \beta^* < 1$ *(see Figure 1).*

A major limitation of open source software is that they are not as usable as PS (e.g. Nichols and Twidale 2003; Frishberg *et al.* 2002). This result is interesting in the sense that it provides a possible explanation for this lack of usability in the commercial version of open source software (i.e. OSS-SS). It shows that OSS-SS vendors do not have an incentive to improve the usability of their software to the level of competing PS. If OSS-SS are as usable as PS then they are forced to compete on Bertrand prices (i.e. at marginal production cost *c*) and end up making zero profits (see *Figure 1*).

Proposition 3*: When the software displays a high level of network benefits (i.e.* $\theta = 1$*), the equilibrium prices for PS are higher than the equilibrium prices for OSS-SS only when the OSS-SS is not as usable as PS (see Figure 1).*

This result is evident in the lower prices of commercial open source desktop operating systems (e.g. Red Hat Linux) in comparison to the price charged by the PS option (e.g. Windows). In these markets, however, there is a strong incentive for the OSS-SS vendor to improve the usability of their product. By doing so it can not only charge higher prices, it will also force the PS vendor out of the software market since the equilibrium price that it can charge is not enough to cover the marginal cost of selling the software (see *Figure 1*).

Analysis of the equilibrium demand (see *Table 4*) also leads to some interesting results. *Figure 2* plots the equilibrium demand for the three types of software against the usability level of OSS-SS.

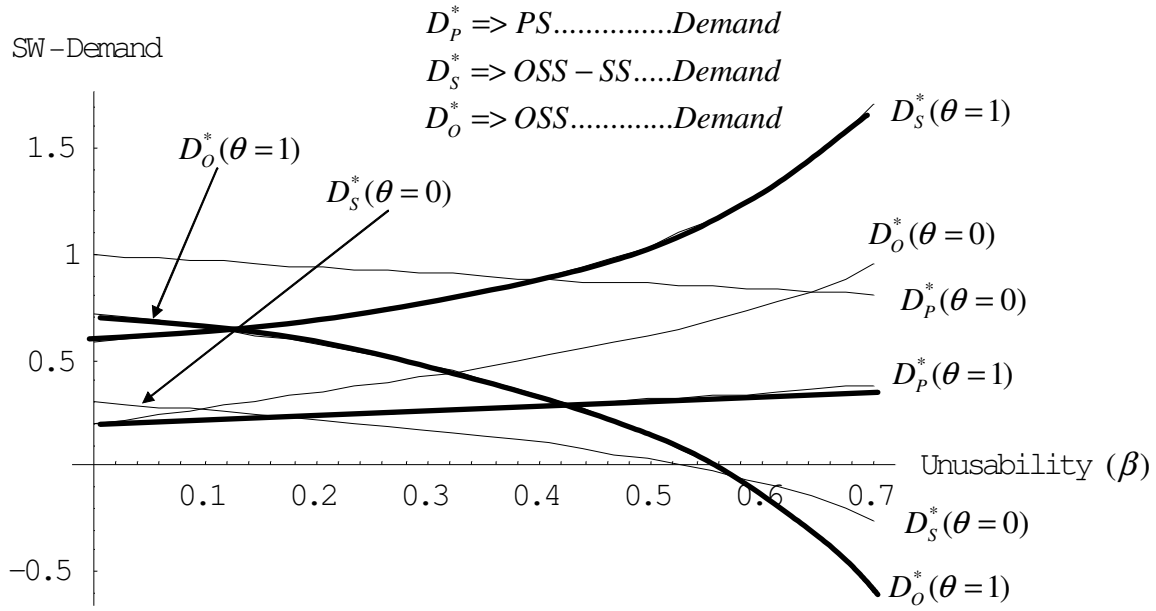

Figure 2: Equilibrium SW Demand $[Assume: V = 1.5; c = 0.2]$

Proposition 4: *When the competing software have weak network benefits* (i.e. $\theta = 0$)*, the market share of PS is always higher than that of OSS-SS (see Figure 2).*

This is an interesting result in light of the fact that an increasing number of firms have already entered or are planning to enter the software market as OSS-SS vendors. *These vendors, however, will never be able to beat the PS vendors in terms of market share* if they are competing in software markets characterized by weak network affects. This result does not imply that PS will always have more market share than its open source alternative. In fact, as the usability of OSS-SS decreases, freely available OSS's share of the market increases. At the same time PS is forced to compete with OSS, which is available for free, resulting in a loss of some market share to OSS. When OSS-SS and OSS are similar in terms of usability, the increase in the market share to OSS is enough to make it the dominant player in the software market (see *Figure 2*). *However, the market*

share of OSS-SS still remains below that of PS. This result provides a possible explanation of why the infrastructure software (e.g. web servers, DNS servers, email servers etc.) market is dominated by OSS such as Apache and BIND. This software market consists of software based on well established communication standards and as a result all potential users benefit from the network affects generated by these software. This could be a possible reason why we do not find many OSS-SS vendors in software categories where the network affects are weak, (e.g. web servers[5], email servers, network servers etc.). The few remaining PS vendors in the infrastructure-software market are fast loosing their once dominant market position to open source alternatives. For instance, Apache enjoys about 69% in web server market while IIS by Microsoft lags far behind with about 21% market share,[6] and in market for DNS servers, the open source option BIND is the predominant name-server on the Internet.

Proposition 5: *When the software market consists of PS, OSS-SS and OSS options, and the software displays strong network benefits (i.e.* $\theta = 1$*), the demand for PS and OSS-SS increase with a decrease in the usability of OSS-SS, while the demand for OSS decreases with a decrease in the usability of OSS-SS (see Figure 2).*

This is a counter intuitive result since one would expect the demand for OSS-SS to decrease with a decrease in its usability. One reason could be that in software markets characterized by strong network benefits, users make their decision on the basis of "expected" demand for each type of software and software usability. PS demand increases because the alternatives (i.e. OSS and OSS-SS) are not as usable. OSS-SS demand increases because it is more usable, and most users expect to get the benefit of

[5] RedHat has entered the web server market with an Apache-based product called Stronghold. However, this product has yet to gain any significant market share.
[6] http://www.serverwatch.com/stats/article.php/3487716

large combined network of OSS-SS and OSS users (even f they do not use the less usable OSS).

Finally, an analysis of the profit functions in *Table 4* results in the following propositions-

Proposition 6a*: In software markets characterized by strong network affects, profits for PS vendor increases with a decrease in the usability of OSS-SS, while that of OSS-SS increases with an increase in its usability (Figure 3a).*

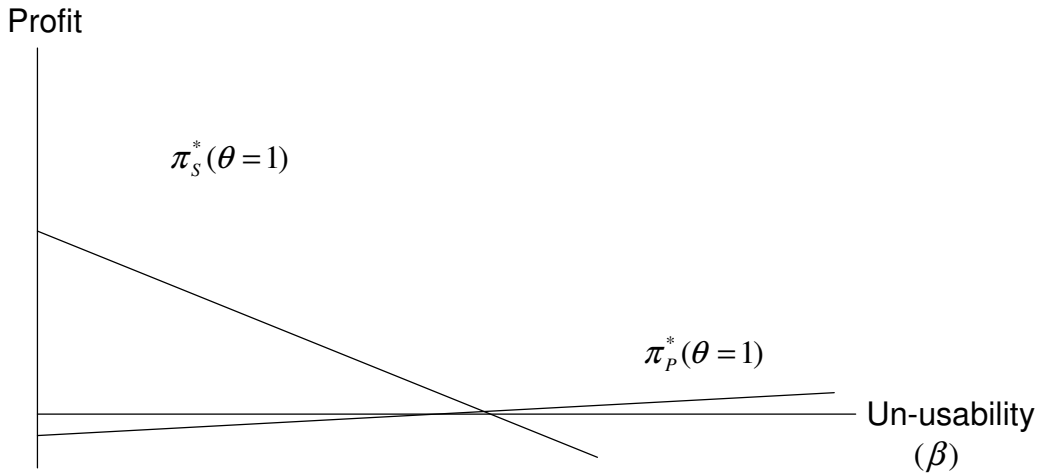

Figure 3a: Equilibrium Profits in Software Markets with High Network Benefits $[Assume: V = 1.5; c = 0.2]$

Proposition 6b*: In software markets characterized by weak network affects, profits for PS vendor increases with a decrease in the usability of OSS-SS, while that of OSS-SS is maximized at* $0 < \beta = \beta_S^* < 1$ *(Figure 3b).*

While *Proposition 6a* is along the expected lines, *Proposition 6b* follows from *Proposition 1*.

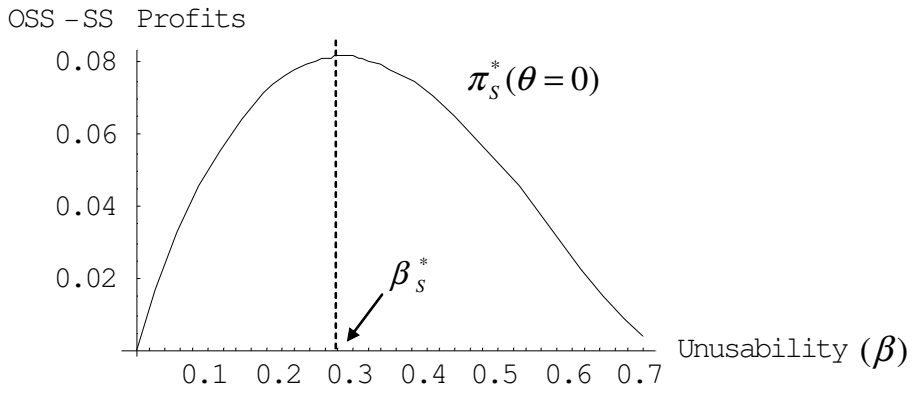

Figure 3b: Equilibrium Profits in Software Markets with Low Network Benefits $[Assume: V = 1.5; c = 0.2]$

4. DISCUSSION

4.1 The Worst Case Scenario for PS and OSS-SS Vendors: The worst case scenario for both the PS and OSS-SS vendors is when OSS is as usable as PS and OSS. In such a scenario there is no incentive for the PS and the OSS-SS vendors to enter the software market. The OSS-SS vendors stay away because they face zero demand (even though they are highly usable) when they compete against *freely available and highly usable OSS.* On the other hand, both PS and OSS-SS will have a positive demand. However, the equilibrium price for PS is not enough to cover the marginal cost of selling the software, resulting in negative profits for the PS vendor (See *Appendix B*). Therefore, both PS and OSS-SS vendors have little incentive to improve the usability of freely available OSS. However, such a scenario begs an important question, i.e. what is the likelihood of this scenario (i.e. the usability of OSS application improves to the extent that it is comparable to that of PS applications) ever happening? For this to happen, the developers involved with the OSS project will have to give as much importance to the usability of the application as to its utility or functionality, and will have to develop user-friendly

interfaces and features for the OSS application. Though theoretically possible, a realistic assessment of OSS projects shows that this is not likely to happen soon. OSS application developers have traditionally favored function over form, which is evident in the lack of user-friendly features (e.g. user interfaces) in these applications and there is no evidence that this is changing (Nichols and Twidale 2003). One reason being that in most open source projects, the developers of OSS applications are also its core users (Mockus *et al.* 2000, 2002). For these *developer-users* the OSS application is already "user-friendly" because they are highly skilled software professionals (Koch 2003), and they already know the software inside out (the source code is available to them) to modify it as they choose fit. In addition, most OSS developers work under a license that prevents them from selling the code, so they develop features that are important to them and not necessarily to the market. Finally, there is hardly any mechanism to capture the feedback of average users who are not involved with the OSS application development process, but are most in need of user-friendly features. Therefore, we can safely assume that the likelihood of the worst case scenario ever happening is low.

4.2 How can commercial proprietary software producers compete against open source software?

This is one of the key questions that we started the paper with. In this paper we divide the software markets into two broad categories- those that are characterized by strong network affects and those that are characterized by weak network affects. A vendor of PS software needs to identify the software market that it is competing in and act accordingly. If it is competing in software markets with strong network benefits, then it will remain an important player in the software market as long as OSS remains highly unusable and

OSS-SS remains unusable relative to the PS (see *Figures 1, 2 and 3a*). This makes the software markets such as e.g. the desktop office productivity software market, and desktop operating systems, attractive for PS vendors. Furthermore, a PS vendor can improve its competitive position by ensuring relatively high usability in their own products. In addition, they need to ensure very low usability in freely available OSS. They can do so by participating in OSS projects, and encouraging the developers of OSS to focus their development efforts only on functionality and reliability of open source software. If the OSS becomes as usable as OSS-SS, then the OSS-SS vendor is forced to improve its own usability (see *Figure 3a*) or exit the market (see *Appendix C*). If the OSS-SS vendor decides to improve its usability further, then all software end up with the same usability resulting in negative profits for PS, and no demand for OSS-SS (See *Appendix B*).

If the PS vendor is competing in software markets characterized by weak network affects, then it should work towards improving the usability of OSS so that it is as usable as OSS-SS. In such a market OSS-SS has two options- (a) to improve its usability in relation to OSS, (b) to exit the market (see *Appendix C*). If OSS-SS decides to improve its usability, it is constrained by the relationship between its usability and profitability (see *Figure 3b*), which ensures that OSS-SS is never as usable as PS (since OSS-SS profits are maximized at $0 < \beta = \beta_S^* < 1$). If possible, the PS vendor should help OSS developers to improve its usability to the level of OSS-SS (i.e. β_S^*). This would force the OSS-SS vendor to exit the software market (see *Appendix C*). However, if the usability of OSS-SS is very low, then the PS vendor is better off by ensuring that OSS has the lowest usability among the three (see *Figure 4*), so that OSS-SS remains in the software market. This is an

interesting result since it identifies the condition under which *the presence of OSS-SS vendor actually helps the competitive position of the PS vendor.*[7]

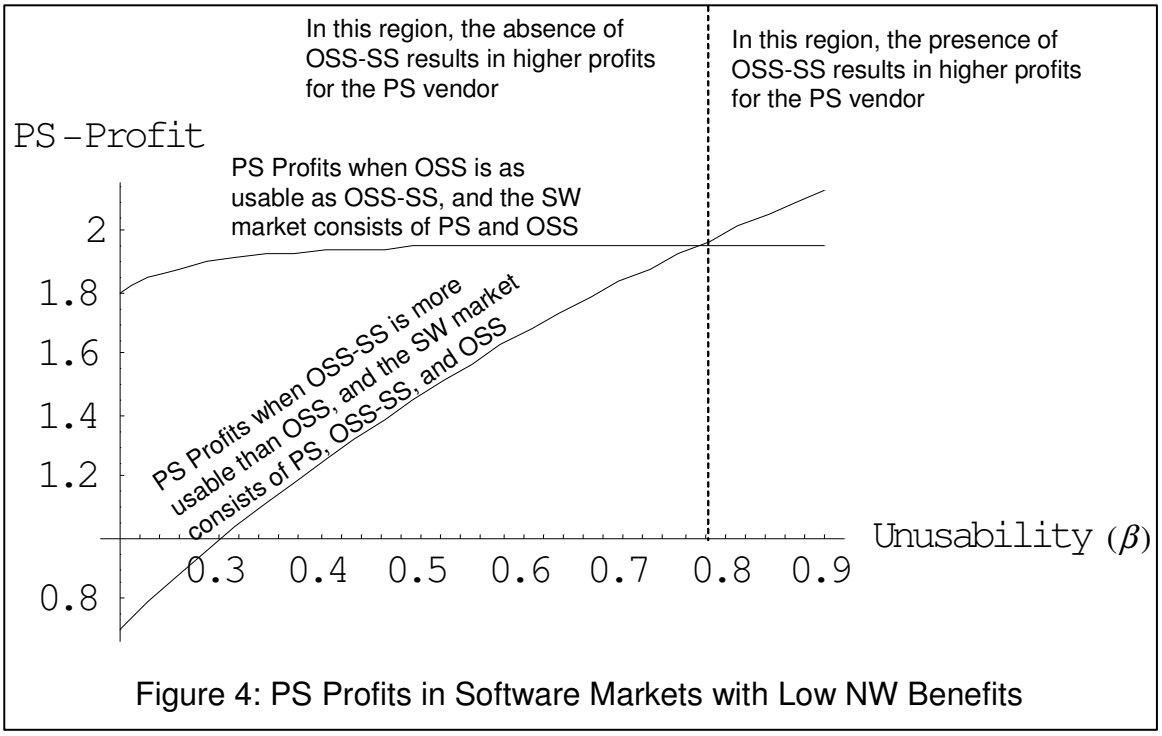

Figure 4: PS Profits in Software Markets with Low NW Benefits

4.3 How can OSS-SS vendors compete against well established proprietary software?

OSS-SS vendors who sell usable OSS plus support services face a very competitive environment. On one hand they have to compete with highly usable PS option, and on the other hand they have to compete with freely available OSS. To compete successfully, the OSS-SS vendor needs to benchmark its usability against the competing PS. If it is competing in software market characterized by weak network affects, then it is better off by ensuring that OSS-SS is highly usable compared to the OSS, but not as usable as the competing PS (see *Figures 1,* and *3b*). On the other hand, if it is competing in software

[7] Some values that we have assumed in our plots (i.e. V=1.5 and c=0.2) result in plots (e.g. Figure 2) which suggest that OSS-SS vendor will never settle for low usability. However, if we assume a lower value for *c* (e.g. c= 0.01), then the OSS-SS will have a positive equilibrium demand and price. As result, there is a realistic chance for the existence of OSS-SS with very low usability (in comparison to PS).

markets characterized by high network affects then it needs offer a product that is at least as usable as the competing PS. By doing so it can force the PS vendor out of the software market (see *Figure 1* and *Figure 3a*). OSS-SS vendors also need to keep in mind that users differentiate between OSS and OSS-SS mainly on the basis of usability. If OSS starts to become more usable, then OSS-SS vendors loose this strategic advantage. Therefore, OSS-SS need to ensure that the "free" open source software is not very usable when compared to the usability of PS or OSS-SS. This can be achieved by actively participating in relevant open source projects and having enough influence among the developers to focus most of their development efforts towards improving software functionality and reliability. In addition, they should *provide a highly usable version of OSS only if the users also buy the support services.* For instance, some OSS-SS vendors continue to provide highly usable versions of relevant OSS for "free" (e.g. Suse 9 by Novell is a highly usable Linux flavor for desktops and it is available for free download). This strategy will have an adverse impact on their market shares and profitability in relevant software markets (see *Appendix B*).

5. CONCLUSION

This paper proposes an analytical model to study the competition between commercial and open source software options. The main results obtained after analysis the competition between open source software and proprietary software application are summarized in *Table 5*.

Table 5: Threat from OSS & OSS-SS, and PS vendor's strategic options

		Strength of NW Affects	
		Scenario #1 (Low)	**Scenario #2 (High)**
Usability of Commercial Open Source Software	**Low**	Open source software dominates PS in terms of market share. In this segment the dominance of non-commercial open source software is evident in infrastructure	Currently, PS has the dominant market share in this software segment. However, we should expect to see the open source software dominating this market

		software segment (e.g. Web Servers, DNS servers, email servers) **Managerial Implications for PS Vendor:** Encourage the entry of OSS-SS with low usability, and ensure that OSS has lower usability than OSS-SS by actively participating in OSS projects.	in future. The strong performance of Linux, MySQL, and increasing awareness about OpenOffice illustrates this trend. **Managerial Implications for PS Vendor:** PS can remain competitive by ensuring that it remains more usable than OSS and OSS-SS.
	High	In this segment PS will dominate in terms of market share but OSS-SS and OSS will have a significant presence. This is evident in the emergence of OSS-SS and OSS as a major alternative to PS in these software markets (e.g. personal firewalls, web browsers, email clients). **Managerial Implications for PS Vendor:** Ensure that OSS is as usable as OSS-SS by actively participating in OSS projects.	We have yet to see a commercial version of open source in this software category that is as usable as the PS. Therefore, we have yet to see any competition in this segment. **Managerial Implications for PS Vendor:** PS can remain competitive by ensuring that it remains more usable than OSS and OSS-SS.

The proposed model is generic in nature because at the end of the day, irrespective of the software category (e.g. desktop application, systems software, enterprise applications etc.), and type of potential users (i.e. individuals, organizations, system administrators, programmers etc.), the users' choices are decided on the basis factors such as their needs (e.g. as suggested by task-technology fit literature); the perceived value offered by various options (e.g. as illustrated by technology adoption models); and total cost of software ownership (e.g. as suggested by transaction cost theory). The model proposed in this paper incorporates these factors. For instance, software category is controlled for by assuming that competing software have the required functionality and reliability, and whether they offer weak or strong network benefits; perceived value from the software is represented by the users' utility functions; and the cost of ownership is represented by the price paid by the potential users. Thus the proposed general utility model captures the

essential decision variables (although in an aggregate form) used by users to make their software choice, and provides us with some significant insights about the impact of this decision on the industrial organization of software industry.

Like most analytical models, the proposed model also has its limitations. One limitation of the model is that PS and OSS are assumed to be compatible with any existing software that may be required to run in conjunction with them. This, however, is not a serious limitation for the simple reason that most open source software are late entrants to the market, which already consists of the PS and its complementary/supplementary products. In order to attract users in this market, the OSS *will* have to be compatible with these existing complementary/supplementary products. This was the case with Linux and most other open source software. In fact Linux now even allows users to emulate Windows on a Linux machine. Another limitation of the model is that the assumptions about users' choice between the proprietary software and open source software are straightforward- i.e., it is influenced by the software price, software usability, and expected installation base. Though this simplicity is important for a flexible but robust model, it is important to acknowledge that users' choice of software is guided by more complex heuristics as suggested in technology adoption literature. Incorporating these to develop a more rich and realistic mathematical model should be undertaken in future research. Finally, the model does not take into account some additional benefits offered by open source software, i.e. access to the software code, and the freedom to modify the software. However, very few people need *direct* access to source code-- only developers or code reviewers need the ability to access and change code. In fact most users do not participate in the development of open source software, and most of the development is done by a

relatively small number of people. In most instances, even the "power users" (e.g. developers, systems administrators) have other full time jobs and responsibilities, and as a result, they do not want to spend time to modify an open source software, or fix a bug in the software, or tailor it to their own unique needs. They are just looking for a software (open source or otherwise) that can perform certain functions without costing them an arm and a leg in terms of price and effort. Therefore, this limitation in the model is not significant in the context of most software users.

Finally a cautionary note- the proposed model variables should be interpreted in the proper context to get meaningful interpretations of the results. For example, even though the model does not differentiate between the type of software (e.g. desktop application and network operating systems), its result should still qualify as long it is applied to study the competition between the same category of software (i.e. they address the same functional needs of the users, display similar network affects).

APENDIX A: Equilibrium Demand, Prices, and Profits

Consumers of type *v* purchase one unit of software that offers them the most utility. This gives the following demand- $D_{PS} = V - \frac{P_P - P_S}{\beta} - \frac{\theta(1-2q_e)}{\beta}$. In a symmetric fulfilled expectation Nash Equilibrium, $q_e = D_{PS}$. Therefore,

$$D_{PS} = V - \frac{P_P - P_S}{\beta} - \frac{\theta(1-2D_{PS})}{\beta} \Rightarrow D_{PS} = \frac{\beta V - P_P + P_S - \theta}{\beta - 2\theta} \quad \textbf{(4a)}$$

$$D_{OSS-SS} = \frac{\theta(1-2V) + P_P}{\beta - 2\theta} - \frac{1-2\theta}{(\beta-2\theta)(1-\beta)} P_S \quad \textbf{(4b)}$$

$$D_{OSS} = \frac{P_S}{1-\beta} \quad \textbf{(4c)}$$

Profit made by PS vendor is:

$$\pi_P = \left[\frac{\beta V - \theta - P_P + P_S}{\beta - 2\theta}\right][P_p - c] \quad \textbf{(5a)}$$

Differentiating π_p w.r.t. P_P, and equating it to zero gives the profit maximizing price, $P_P = \frac{1}{2}[\beta V + c - \theta + P_S]$ **(5b)**

Similarly, the demand for OSS-SS is $\frac{\theta(1-2V) + P_P}{\beta - 2\theta} - \frac{1-2\theta}{(\beta-2\theta)(1-\beta)} P_S$, and profit made by the OSS-SS vendor is –

$$\pi_S = \left[\frac{\theta(1-2V) + P_P}{\beta - 2\theta} - \frac{1-2\theta}{(\beta-2\theta)(1-\beta)} P_S\right][P_S - (1+\beta)c] \quad \textbf{(6a)}$$

Differentiating π_d w.r.t. P_d, and equating it to zero gives the profit maximizing price-

$$P_S = \frac{1}{2(1-2\theta)}\left[\theta(1-\beta)(1-2V) + (1+\beta)(1-2\theta)c + (1-\beta)P_P\right] \quad \textbf{(6b)}$$

Solving for P_P and P_S from Eqs. (2b) and (3b) gives the optimal prices for PS and OSS-SS as follows-

$$P_P^* = \left(\frac{2(1-2\theta)(\beta V - \theta) + (1-2\theta)(3+\beta)c + \theta(1-\beta)(1-2V)}{3+\beta-8\theta} \right) \quad \textbf{(7a)}$$

$$P_S^* = \left(\frac{2\theta(1-\beta)(1-2V) + (1-\beta)(\beta V - \theta) + [3+\beta+4\theta(1+\beta)]c}{3+\beta-8\theta} \right) \quad \textbf{(7b)}$$

Equilibrium demand for OSS, OSS-SS and PS respectively are as follows-

$$D_P^* = \left[\left(\frac{\beta V - \theta}{\beta - 2\theta} \right) - \left(\frac{(1+\beta-4\theta)(\beta V - \theta) - \theta(1-\beta)(1-2V) - 2\theta(5+3\beta)c}{(\beta-2\theta)(3+\beta-8\theta)} \right) \right] \quad \textbf{(8a)}$$

$$D_S^* = \left(\frac{\theta(1-2V)}{(\beta-2\theta)} + \frac{(1+\beta-4\theta)(\beta V - \theta) - \theta(1-\beta)(1-2V) - 2\theta(5+3\beta)c}{(\beta-2\theta)(3+\beta-8\theta)} \right) - \left(\frac{2\theta(1-2V) + (\beta V - \theta)}{3+\beta-8\theta} \right) - \left(\frac{3+\beta+4\theta(1+\beta)}{(3+\beta-8\theta)(1-\beta)} \right) c \quad \textbf{(8b)}$$

$$D_O^* = \left(\frac{2\theta(1-2V) + (\beta V - \theta)}{3+\beta-8\theta} \right) + \left(\frac{3+\beta+4\theta(1+\beta)}{(3+\beta-8\theta)(1-\beta)} \right) c \quad \textbf{(8c)}$$

APPENDIX B: Competition when OSS is as usable as PS and OSS-SS

The utility functions for consumers are as follows:

$U_P(v) = A + \theta q - v - P_P$

$U_S(v) == A + \theta(1-q) - v - P_S$

$U_O(v) = A + \theta(1-q) - v$

Consumers of type v purchase one unit of software that offers them the most utility. Therefore, OSS-SS has zero demand since users always prefer OSS over it. However, a consumer of type v will prefer PS over OSS if - $P_P \le \theta(2q_e - 1)$. Therefore, the profit maximizing price for PS vendor is $P_P = \theta(2q_e - 1)$, and the profit function is given

by $\pi_P = D_P[\theta(2q_e - 1) - c]$. In a symmetric fulfilled expectation Nash Equilibrium $D_P = q_e = q$, and the profits are maximized at $q^* = \frac{\theta + c}{4\theta}$. Equilibrium price is given by $P_P^* = \frac{1}{2}(c - \theta)$. Since $P_P^* = \frac{1}{2}(c - \theta) < c$, the PS firm does not make any profit.

APPENDIX C: Competition when OSS is as usable as OSS-SS

The utility functions for consumers are as follows:

$U_P(v) = A + \theta q - v - P_P$

$U_S(v) == A + \theta(1 - q) - (1 + \beta)v - P_S$

$U_O(v) = A + \theta(1 - q) - (1 + \beta)v$

Consumers of type v purchase one unit of software that offers them the most utility. Therefore, OSS-SS has zero demand since users always prefer OSS over it. However, a consumer of type v will prefer PS over OSS if - $v > \frac{\theta(1 - 2q_e) + P_P}{\beta}$. Since, in a symmetric fulfilled expectation Nash Equilibrium, the expected demand is same as the actual demand, $D_P = q_e = q \Rightarrow q = V - v = V - \frac{\theta(1 - 2q) + P_P}{\beta} \Rightarrow q = \frac{\beta V - \theta - P_P}{\beta - 2\theta}$. The profit function for PS vendor is given by $\pi_P = D_P(P_P - c) = \left(\frac{\beta V - \theta - P_P}{\beta - 2\theta}\right)(P_P - c)$. This profit is maximized at $P_P^* = \frac{1}{2}(\beta V + c - \theta)$. The equilibrium demand is given by $D_P^* = \frac{\beta V - \theta - c}{2(\beta - 2\theta)}$, and the equilibrium profits are $\pi_P^* = \left[\frac{\beta V - \theta - c}{2(\beta - 2\theta)}\right]\left[\frac{\beta V + c - \theta}{2(\beta - 2\theta)} - c\right]$.

Thus, as long as $c < \beta V - \theta$, PS vendors can charge a price higher than marginal production cost, will enjoy positive demand and a positive profit.

References:

1. Adams, D.A., Nelson, R.R., and Todd, P.A. 1992. Perceived Usefulness, Ease of Use, and Usage of Information Technology: A Replication. *MIS Quarterly,* (16:2), June, 227-247).
2. Bessen, J. 2002. What Good is Free Good? In Hahn ed., *Government Policy Towards F/OS Software*. AEI Brooking Joint Center for Regulatory Studies, Washington D.C.
3. Davis, F.D. 1989. Perceived Usefulness, Perceived Ease of Use and User Acceptance of Information Technology. *MIS Quarterly* (13:3), September, 319-340.
4. Farrell, J., and Saloner, G. 1985. Standardization, Compatibility, and Innovation. *Rand Journal of Economics*, (16), 70-83.
5. Farrell, J., and Saloner, G. 1986. Installed Base and Compatibility: Innovation, Product Pronouncements, and Predation. *American Economic Review*, (76), 1986, 940-955.
6. Frishberg, N., Dirks, A. M., Benson, C., Nickell, S., and Smith, S. 2002. Getting to Know You: Open Source Development Meets Usability. *CHI 2002, Minneapolis, MI,* ACM Press, 2002.
7. Gaudeul, A. 2003. The $(L^A)T_E X$ Project: A Case Study of Open-Source Software. *TUGBoat*, (24), 1001-1015.
8. Green, E. L. Economics of Open Source. *http://badtux.org/home/eric/editorial/economics.php*
9. Hartwick, J. and Barki, H. 1994. Explaining the Role of User Participation in Information System Use. *Management Science* (40:4), December, 440-465.

10. Johnson, J.P. 2002. Economics of Open Source Software. *Journal of Economics and Management Strategy,* 11(4), 637-662.
11. Karahanna, E., Straub, D.W., and Chervany, N.L. 1999. Information Technology Adoption Across Time: A Cross-Sectional Comparison of Pre-adoption and Post-adoption Beliefs. *MIS Quarterly* 23(2), June, 183-213.
12. Katz, M. and Shapiro, C. 1999. Antitrust in Software Markets. *Competition, Innovation and the Microsoft Monopoly: Antitrust in the Digital Marketplace,* Eisenach, J. and Lenard, T. (eds.), Kluwer Academic Publishers: Boston.
13. Katz, M. and Shapiro, C. 1986. Technology Adoption in the Presence of Network Externalities. *Journal of Political Economy,* (94), 822-841.
14. Koch, C. 2003. Your Open Source Plan. *CIO*, March 15.
15. Lederer, A. L., Maupin, D.J., Dena, M.P., and Zhuang, Y. 2000. The Technology Acceptance Model and The World Wide Web. *Decision Support Systems* 29(3), October, 269-282.
16. Lerner, J. and Tirole, J. 2002. Some Simple Economics of Open Source. *Journal of Industrial Economics*, (50), 197-234.
17. Manes, S. 2002. Linux Gets Friendlier. *Forbes*, 10 June, 134-136.
18. Mockus, A., Fielding, R., and Herbsleb, J. 2000. A Case Study of Open Source Software: The Apache Server. in *Proceedings of 22nd International Conference on Software Engineering,* Limmerick, IR, 263-272.
19. Moon, J.Y., and Sproull, L. 2000. Essence of Distributed Work: The Case of the Linux Kernel. *Firstmonday*, 5(11).

20. Moore, G.C. and Benbasat, I. 1991. Development of an Instrument to Measure The Perceptions of Adopting an Information Technology Innovation. *Information System Research*, 2(3), September, 192-222.

21. Mustonen, M., 2003. Copyleft- The Economics of Linux and Other Open Source Software. *Information and Economics Policy*, 15(1), 99-121.

22. Nichols, D. M., and Twidale, M. B. 2003. The Usability of Open Source Software. *First Monday*, 8(1), January.

23. Nielsen, J. 1993. *Usability Engineering*, Boston: Academic Press.

24. Raymond, Eric S. 1999. *The Cathedral and Bazaar*, O'Reilly: Sebastopol, California.

25. Rogers, E.M. 1995. *Diffusion of Innovations*, New York: The Free Press.

26. Tirole, J. 2000. *The Theory of Industrial Organization*. Cambridge, MA: The MIT Press.

27. Venkatesh, V. and Davis, F.D. 2000. Theoretical Extension of The Technology Acceptance Model: Four Longitudinal Field Studies. *Management Science*. 46(2), February, 186-204.

28. Venkatesh, V. and Morris, M.G. 2000. Why Don't Men Ever Stop to Ask For Directions. Gender, social influence, and their role in technology acceptance and usage behavior. *MIS Quarterly*, 24(1), March, 115-139.

29. Viega, J. 2004. Open Source Security: Still a Myth. *www.onlamp.com/pub/a/security/2004/09/16/open_source_security_myths.html*.