OmniaScience
Journal of Industrial Engineering and Management
JIEM, 2015 – 8(3): 861-876 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423
http://dx.doi.org/10.3926/jiem.1362

The Effect of Competition from Open Source Software on the Quality of Proprietary Software in the Presence of Network Externalities

Mingqing Xing

School of Economics and Management, and Neural Decision Science Laboratory, Weifang University (China)

mqxing1979@163.com

Received: December 2014
Accepted: May 2015

Abstract:

Purpose: A growing number of open source software emerges in many segments of the software market. In addition, software products usually exhibit network externalities. The purpose of this paper is to study the impact of open source software on the quality strategies of proprietary software vendors when the market presents positive network externalities.

Design/methodology/approach: To analyze how open source software affects the quality of proprietary software, this paper constructs two vertical differentiation models: the basic model considers proprietary software monopolizing the market, and the extended model considers proprietary software competing with open source substitute.

Findings: This paper mainly finds that the presence of open source software does not necessarily lead to the improvement of proprietary software quality. The network externalities and the compatibility between open source and proprietary software can change the impact of open source software on the quality of proprietary software and affect the quality strategies of proprietary software vendors under certain conditions.

Originality/value: The main contribution of this paper is to examine the effect of open source software on the quality strategies for proprietary software vendors in software markets with network externalities.

Keywords: quality strategy, open source software, proprietary software, functional quality, software usability, network externality, compatibility

1. Introduction

The rapid development of open source software (OSS) is one of the most important events in the software industry. Open source software gives developers the freedom to run it for any purpose, to share its source codes, to identify and correct errors, and to redistribute its source codes (O'Reilly's, 1999; Wheeler, 2007). It has gradually become a great threat to proprietary software in many markets. For examples, Linux, as an open source operating system, occupies more than thirty percent market share in the server operating system market, and Microsoft's Windows, as a proprietary operating system, holds about fifty percent market share; over three-fifths of websites adopt Apache (an open source software) in the web server market, but only around three-tenths use Microsoft's Internet Information (a proprietary software) (Lin, 2008).

Some scholars study the competition between proprietary software and its open source alternatives. Dalle and Jullien (2002) examine the technological competition between proprietary and open source software by an interaction theory model; Meng and Lee (2005) investigate the compatibility between open source software and its proprietary substitute; Lin (2008) analyzes the impact of users' expertise on the market in which proprietary software competes with open source software; Pradniwat (2008) and Gramstad (2014) study the competition between a commercial software and a free-of-charge open source substitute in the emergence of piracy; Xing (2010) considers the quantity competition between open source and proprietary software providers; Cheng, Liu and Tang (2011) examine how the network externalities affect the competition between proprietary and open source software; Xing (2013, 2014a) investigates the price competition between commercial open source and proprietary software; Zeroukhi and Pénard (2014) study whether public subsidies for open source software are socially desirable and how the extent of compatibility between open source software and proprietary software influences the optimal subsidy offered. However, they do not consider the impact of open source software on the quality of proprietary software.

As exceptions, Raghunathan, Prasad, Mishra and Chang (2005) compare the optimal quality under proprietary software monopoly and duopolistic competition between proprietary and open source software, and the result shows that both open source and proprietary qualities decrease in a competitive market; Lanzi (2009) analyzes the quality competition between closed and open source software, and argues that, in comparison to the monopolistic case, the quality of proprietary software increases in a duopoly created by the presence of open source software; Choudhary and Zhou (2007) examine the influence of competition from open source software on the optimal quality of proprietary software, and the result suggests that the

quality developed by proprietary vendors under duopoly competition may be higher or lower than monopoly quality; Xing (2012) investigates how the quality of open source software impacts proprietary software's quality, and the result shows that the quality of proprietary software may not increase with the increase of open source software's quality. However, they do not consider whether the network externalities and the compatibility between open source and proprietary software can change the impact of open source software on proprietary software's quality. Compared with traditional physical goods, one of the most notable characteristics is that software products usually exhibit positive network externalities, which refer to the increase of user utility when more users employ the same or compatible products (Katz & Shapiro, 1985; Shy, 2001). The positive network externalities effect of software may promote a software product enjoying a commanding market share (Cheng et al., 2011). To win in the battle of market share, proprietary software vendors may strategically choose to improve (or reduce) their product quality when compete with open source software. Consequently, the network externalities and compatibility may change the influence of open source software on proprietary vendors' quality strategies. This study examines how open source software affects proprietary software's quality in a market with network externalities. Two theoretical models are established by extending the vertical differentiation setting of Mussa and Rosen (1978). Their model assumes that the preference heterogeneity of consumers is one dimensional, and studies the monopoly quality choice. This paper mainly finds that: (i) the appearance of open source software does not necessarily lead to the higher quality of proprietary software; (ii) the network externalities and software compatibility can change the impact of open source software on proprietary software's quality, and affect the quality choices for proprietary vendors under certain conditions.

The rest of this study is organized as follows. Section 2 presents the basic model and solves the optimal quality of proprietary software when it monopolizes the market. Section 3 analyzes the case when proprietary software competes with open source software. Section 4 compares the optimal qualities for proprietary software in section 2 and section 3. The final section concludes this study.

2. The Basic Model

Consider a software market which exhibits positive network externalities. Potential software users are indexed by their level of the technical skills, which is measured by θ. Assume that the software users with higher level of technical ability have lower θ, but those with lower level of technical ability have higher θ. A user's willingness to pay for the software usability is higher when his/her level of technical skills is lower (Choudhary & Zhou, 2007). The total number of potential users is normalized to 1 and users are uniformly distributed over the interval [0,1]. A proprietary software vendor monopolizes the market and open source software does not

appear in the basic model. The utility for the potential user at $\theta \in [0,1]$ when he/she adopts proprietary software is defined as:

$$u_p = \theta v_p + f_p + \alpha d_p - p_p \tag{1}$$

In (1), v_p and f_p are the usability and functional quality of proprietary software respectively; p_p is the price for proprietary software; αd_p presents the utility user deriving from the network externalities, in which α ($\alpha > 0$) is the intensity of network externalities and d_p is the installed base of proprietary software (Katz & Shapiro, 1985; Xing, 2014b). Note that: (i) the software quality is assumed to only depend on its usability (includes the ease of installation and user interface, the level of technical support, etc) and functional quality (includes feature set, reliability, security, etc) (Choudhary & Zhou, 2007); (ii) this study only considers a direct network externalities effect (Shy, 2001).

The profit function of proprietary vendor is given by:

$$\pi_p = d_p p_p - r f_p^2 \tag{2}$$

In (2), rf_p^2 denotes the cost of improving functional quality, where r is a positive parameter.

The proprietary vendor decides the functional quality (f_p) and price (p_p) for its software. Note that the usability of proprietary software (v_p) is assumed to be exogenous. The timing of choosing quality and pricing is as follows. In the first stage, the proprietary vendor determines the functional quality of software. In the second stage, it sets software price. According to the market coverage, the following two cases are considered.

2.1. Case I: Proprietary Software Monopolizes in a Fully Covered Market

If the market is fully covered, all potential users adopt the proprietary software. In this case, the installed base equals 1 (i.e., $d_p = 1$). The proprietary vendor chooses its price according to the equality: $f_p + \alpha - p_p = 0$. Thus, the optimal price for proprietary software is:

$$p_P^{\#} = \alpha + f_p \tag{3}$$

Then the profit function of proprietary vendor on f_p is given by:

$$\pi_p = \alpha + f_p - r f_p^2 \tag{4}$$

Solving the first order condition of profit function gives the optimal functional quality:

$$f_p^{\#} = \frac{1}{2r} \tag{5}$$

Obviously, the second order condition is met when $r > 0$. In this case, the optimal functional quality of proprietary software does not depend on the network externalities.

2.2. Case II: Proprietary Software Monopolizes in a Partially Covered Market

If the market is partially covered, some potential users with high level of technical skills do not adopt the proprietary software. In this case, the marginal user type ($\bar{\theta}$) who is indifferent between adopting and not adopting proprietary software is given by: $\bar{\theta} v_p + f_p + \alpha d_p - p_p = 0$. Solving this equation yields:

$$\bar{\theta} = \frac{p_p - f_p - \alpha d_p}{v_p} \tag{6}$$

Since the installed base meets: $d_p = 1 - \bar{\theta}$, the demand function of proprietary software is:

$$d_p = \frac{v_p - p_p + f_p}{v_p - \alpha} \tag{7}$$

Thus, the profit function of proprietary vendor is:

$$\pi_p = \frac{p_p(v_p - p_p + f_p)}{v_p - \alpha} - r f_p^{\,2} \tag{8}$$

According to the first order condition, the optimal price of proprietary software is:

$$p_p^{\&} = \frac{v_p + f_p}{2} \tag{9}$$

The second order condition requires: $v_p > \alpha$. The profit function of proprietary vendor on f_p is given by:

$$\pi_p = \frac{(v_p + f_p)^2}{4(v_p - \alpha)} - r f_p^{\,2} \tag{10}$$

The first order condition of (10) with respect to f_p gives the optimal functional quality of proprietary software:

$$f_p^{\&} = \frac{v_p}{4r(v_p - \alpha) - 1} \tag{11}$$

The second order condition requires:

$$r > \frac{1}{4(v_p - \alpha)} \tag{12}$$

Obviously, the optimal functional quality of proprietary software increases as the intensity of network externalities increases when the proprietary vendor partially monopolizes the market. However, it does not depend on the network externalities when the proprietary vendor monopolizes the whole market.

3. Case III: Proprietary Software Competes with Open Source Software

This section considers the case that open source software also presents in the market. Open source software is from a not-for-profit community. In this case, proprietary software competes with its open source substitute. The utilities for the potential user at $\theta \in [0,1]$ when he/she adopts open source and proprietary software are respectively defined as:

$$u_o = \theta v_o + f_o + \alpha(d_o + kd_p) - p_o \tag{13}$$

$$u_p = \theta v_p + f_p + \alpha(d_p + kd_o) - p_p \tag{14}$$

In (13) and (14), v_i, f_i, p_i and d_i are the usability, functional quality, price and installed base for software i respectively, where $i = o, p$; $k(0 \le k \le 1)$ is the degree of compatibility between proprietary and open source software. The subscript 'o' denotes open source software and the subscript 'p' denotes proprietary software. Note that: (i) since open source software can be freely available from the open source community, its price is equal to zero (i.e., $p_o = 0$); (ii) open source software's main weakness is the low usability and It is generally less user-friendly than its proprietary software substitutes (CIO, 2002). This study supposes that the usability of open source software is inferior to proprietary software (i.e., $v_o < v_p$); (iii) the usability of proprietary software (v_p) and the quality of open source software (v_o and f_o) are assumed to be exogenous.

According to (13) and (14), the marginal user who is indifferent between employing open source and proprietary software ($\tilde{\theta}$) is given by $u_o = u_p$:

$$\tilde{\theta} v_o + f_o + \alpha(d_o + kd_p) = \tilde{\theta} v_p + f_p + \alpha(d_p + kd_o) - p_p \tag{15}$$

Solving (15) gives:

$$\tilde{\theta} = \frac{p_p - (f_p - f_o) - \alpha(1-k)(d_p - d_o)}{v_p - v_o} \tag{16}$$

When open source software appears in the market, the users with high level of technical skills will adopt open source software. Assume that the market is fully covered in the case that proprietary software competes with open source software.

Thus,

$$d_o = \tilde{\theta} \tag{17}$$

$$d_p = 1 - \tilde{\theta} \tag{18}$$

According to (16), (17) and (18), the demand functions for open source and proprietary software are respectively given by:

$$d_o = \frac{(p_p - f_p + f_o)[v_{po} - 2\alpha(1-k)] - \alpha(1-k)(v_{po} - 2p_p + 2f_p - 2f_o)}{v_{po}[v_{po} - 2\alpha(1-k)]} \tag{19}$$

$$d_p = \frac{(v_{po} - p_p + f_p - f_o)[v_{po} - 2\alpha(1-k)] + \alpha(1-k)(v_{po} - 2p_p + 2f_p - 2f_o)}{v_{po}[v_{po} - 2\alpha(1-k)]} \tag{20}$$

Where $v_{po} = v_p - v_o$. Thus the profit functions for open source community and proprietary vendor are:

$$\pi_o = 0 \tag{21}$$

$$\pi_p = \frac{p_p\{(v_{po} - p_p + f_p - f_o)[v_{po} - 2\alpha(1-k)] + \alpha(1-k)(v_{po} - 2p_p + 2f_p - 2f_o)\}}{v_{po}[v_{po} - 2\alpha(1-k)]} - rf_p^{\,2} \tag{22}$$

In this section, the timing of choosing quality and pricing is as follows: the proprietary vendor decides the functional quality of its software in the first stage; the proprietary vendor and open source community price their software in the second stage. The model is solved by backwards induction. The price stage is firstly analyzed and then the quality stage is studied.

In the second stage, software providers set price. Since the price of open source software is equal to zero (i.e., $p_o = 0$), only the optimal price for proprietary software needs to solve. The first order condition of (22) with respect to p_p yields:

$$p_p^* = \frac{v_{po} + f_p - f_o - \alpha(1-k)}{2} \tag{23}$$

The second order condition requires: $v_{po} - 2\alpha(1 - k) > 0$. Plugging (23) into (22), the profit function of proprietary vendor on f_p is:

$$\pi_p = \frac{[v_{po} - \alpha(1-k) + f_p - f_o]^2}{4[v_{po} - 2\alpha(1-k)]} - rf_p^{\,2} \tag{24}$$

In the first stage, the proprietary vendor chooses its software quality. The first order condition of (24) with respect to f_p is:

$$\frac{\partial \pi_p}{\partial f_p} = \frac{v_{po} - \alpha(1-k) + f_p - f_o}{2[v_{po} - 2\alpha(1-k)]} - 2rf_p = 0 \tag{25}$$

Solving (25) gives the optimal quality of proprietary software:

$$f_p^* = \frac{v_{po} - \alpha(1-k) - f_o}{4r[v_{po} - 2\alpha(1-k)] - 1} \tag{26}$$

The second order condition requires:

$$r > \frac{1}{4[v_{po} - 2\alpha(1-k)]} \tag{27}$$

In order to guarantee the positive demands for both open source and proprietary software, the following condition must hold:

$$\frac{1 + 6r\alpha(1-k) - 2rv_{po}}{2r} < f_o < v_{po} - \alpha(1-k) \tag{28}$$

Proposition 1. (i) when $0 \le k < 1$, f_p^* increases with α if $f_o < \frac{4rv_{po}+1}{8r}$, while it deceases with α if $f_o > \frac{4rv_{po}+1}{8r}$; (ii) when $k = 1$, f_p^* is not affected by α.

Proof. according to (26), $\frac{\partial f_p^*}{\partial \alpha} = \frac{(1-k)(4rv_{po}+1-8rf_o)}{\{4r[v_{po}-2\alpha(1-k)]-1\}^2}$.

Thus, $\frac{\partial f_p^*}{\partial \alpha} > 0$ if $0 \le k < 1$ and $f_o < \frac{4rv_{po}+1}{8r}$, $\frac{\partial f_p^*}{\partial \alpha} < 0$ if $0 \le k < 1$ and $f_o > \frac{4rv_{po}+1}{8r}$, and $\frac{\partial f_p^*}{\partial \alpha} = 0$ if $k = 1$.

The above proposition shows that, when proprietary software and open source software are not fully compatible ($0 \le k < 1$), the optimal functional quality of proprietary software increases as the intensity of network externalities increases if the functional quality of open source software is sufficiently low, while the opposite may occur if the functional quality of open source software is sufficiently high. However, when proprietary software is fully compatible with open source software ($k = 1$), the optimal functional quality of proprietary software is not affected by the network externalities. It is worth noting that the impact of network externalities on the optimal functional quality of proprietary software may depend on the degree of compatibility between open source and proprietary software and the quality of open source software. Moreover, the functional quality of proprietary software may decrease with the intensity of network externalities. In contrast to the case of proprietary software monopolizing in section 2, the impact of network externalities on the quality of proprietary

software may differ. That is, the presence of open source software may change the impact of network externalities on the quality choices of proprietary vendor.

Proposition 2. (i) when $f_o < \frac{4rv_{po}+1}{8r}$, f_p^* deceases with k; (ii) when $f_o > \frac{4rv_{po}+1}{8r}$, f_p^* increases with k.

Proof. according to (26), $\frac{\partial f_p^*}{\partial k} = -\frac{\alpha(4rv_{po}+1-8rf_o)}{\{4r[v_{po}-2\alpha(1-k)]-1\}^2}$.

Thus, $\frac{\partial f_p^*}{\partial k} < 0$ if $f_o < \frac{4rv_{po}+1}{8r}$, while $\frac{\partial f_p^*}{\partial k} > 0$ if $f_o > \frac{4rv_{po}+1}{8r}$.

Proposition 2 indicates that, when the functional quality of open source software is low enough, the optimal functional quality of proprietary software decreases as the degree of compatibility between proprietary and open source software increases, while when the functional quality of open source software is high enough, the opposite situation may arise. Note that: (i) the degree of compatibility between proprietary and open source software may affect the quality strategies of proprietary vendor; (ii) the impact of software compatibility on the functional quality of proprietary software may depend on the quality of open source software.

Proposition 3. (i) when $f_o < \frac{1}{8r} + \frac{v_{po}}{2}$, f_p^* is higher if $k = 0$ than if $k = 1$; (ii) when $f_o > \frac{1}{8r} + \frac{v_{po}}{2}$, f_p^* is lower if $k = 0$ than if $k = 1$.

Proof. according to (26), $f_p^*\big|_{k=0} = \frac{v_{po}-\alpha-f_o}{4r(v_{po}-2\alpha)-1}$ and $f_p^*\big|_{k=1} = \frac{v_{po}-f_o}{4rv_{po}-1}$.

When $f_o < \frac{1}{8r} + \frac{v_{po}}{2}$, $f_p^*\big|_{k=0} = \frac{v_{po}-\alpha-f_o}{4r(v_{po}-2\alpha)-1} > \frac{v_{po}-f_o}{4rv_{po}-1} = f_p^*\big|_{k=1}$.

While when $f_o > \frac{1}{8r} + \frac{v_{po}}{2}$, $f_p^*\big|_{k=0} = \frac{v_{po}-\alpha-f_o}{4r(v_{po}-2\alpha)-1} < \frac{v_{po}-f_o}{4rv_{po}-1} = f_p^*\big|_{k=1}$.

The above proposition gives a comparison of proprietary software's optimal functional quality under two specific compatible degrees ($k = 0$ and $k = 1$). It shows that the optimal functional quality of proprietary software when two types of software are fully compatible may be higher or lower than when they are fully incompatible.

Proposition 4. (i) f_p^* decreases with f_o; (ii) when $f_o < \frac{4r\alpha(1-k)+1}{4r}$, f_p^* increases with v_o; when $f_o > \frac{4r\alpha(1-k)+1}{4r}$, f_p^* decreases with v_o.

Proof. according to (26), we can prove $\frac{\partial f_p^*}{\partial f_o} = -\frac{1}{4r[v_{po} - 2\alpha(1-k)]-1} < 0$ and $\frac{\partial f_p^*}{\partial v_o} = -\frac{4rf_o - 4r\alpha(1-k)-1}{\{4r[v_{po} - 2\alpha(1-k)]-1\}^2}$. Thus, $\frac{\partial f_p^*}{\partial v_o} > 0$ if $f_o < \frac{4r\alpha(1-k)+1}{4r}$, and $\frac{\partial f_p^*}{\partial v_o} < 0$ if $f_o > \frac{4r\alpha(1-k)+1}{4r}$.

Proposition 4 demonstrates that, the optimal functional quality of proprietary software decreases as the functional quality of open source software increases. Moreover, it may also be affected by the usability of open source software. If the functional quality of open source software is sufficiently low, the optimal functional quality of proprietary software increases as the usability of open source software increases, while if the functional quality of open source software is sufficiently high, the opposite may appear. We find similar conclusions in another paper (Xing, 2012), which does not consider the network externalities and software compatibility. It is worth noting that the impact of the functional quality and usability of open source software on the functional quality of proprietary software may be different. Consequently we need to distinguish the different quality characteristics of open source software when examine how open source software influences the quality strategies of proprietary vendors. However, most of the relevant research has no distinction between the functional quality and usability of open source software (Raghunathan et al., 2005; Lanzi, 2009).

4. Comparison

This section compares the optimal results under the case of proprietary software monopolizing and that under the case of it facing the competition of open source software.

4.1. Comparison of $f_p^{\#}$ and f_p^{*}

Comparing the optimal functional quality of proprietary software under case I and that under case III, the following result is find.

Proposition 5. $f_p^{\#} > f_p^{*}$.

Proof. according to (28), f_o must meet $\frac{1+6r\alpha(1-k)-2rv_{po}}{2r} < f_o < v_{po}-\alpha(1-k)$.

When $f_o > \frac{1+6r\alpha(1-k)-2rv_{po}}{2r}$, $\frac{v_{po}-\alpha(1-k)-f_o}{4r[v_{po}-2\alpha(1-k)]-1} < \frac{1}{2r}$ holds. Thus, $f_p^{\#} > f_p^{*}$.

The above proposition shows that, proprietary software's optimal functional quality when it monopolizes the whole market is higher than when it competes with open source software. The reason is that, proprietary vendor can obtain more profit when monopolizing the whole market than when facing competition from open source software, thus it has more money to improve its software quality under case I than under case III.

4.2. Comparison of $f_p^{\&}$ and f_p^{*}

This part compares the optimal functional quality of proprietary software under case II and that under case III. Setting $\frac{[v_{po}-\alpha(1-k)][4r(v_p-\alpha)-1]-v_p\{4r[v_{po}-2\alpha(1-k)]-1\}}{4r(v_p-\alpha)-1} = \xi$, the following conclusion can be proven.

Proposition 6. (i) when $f_o < \xi$, $f_p^{\&} < f_p^{*}$; (ii) when $f_o > \xi$, $f_p^{\&} > f_p^{*}$.

Proof. If $f_o < \xi$, $\frac{v_{po}-\alpha(1-k)-f_o}{4r[v_{po}-2\alpha(1-k)]-1} > \frac{v_p}{4r(v_p-\alpha)-1}$. If $f_o > \xi$, $\frac{v_{po}-\alpha(1-k)-f_o}{4r[v_{po}-2\alpha(1-k)]-1} < \frac{v_p}{4r(v_p-\alpha)-1}$. Thus, $f_p^{\&} < f_p^*$ when $f_o < \xi$, and $f_p^{\&} > f_p^*$ when $f_o > \xi$.

Proposition 6 indicates that, when the functional quality of open source software is low enough, proprietary vendor's optimal functional quality under case II is lower than under case III, while the opposite may appear when the functional quality of open source software is high enough. That is, the presence of open source software may lead to the functional quality of proprietary software increase or decrease. This is different from some existing research results (Raghunathan et al., 2005; Lanzi, 2009). In contrast to a proprietary software monopolizing case, Raghunathan et al. (2005) find that the quality of proprietary software will decrease and while Lanzi (2009) finds that the quality of proprietary software will increase when open source software appears in the market. Note that the results of Proposition 6 are different from Proposition 5. The reason is that, the profit of proprietary vendor is far less under case II than under Case I. Under certain conditions, the proprietary vendor can obtain more profit under Case III than under Case II. Thus, the proprietary vendor may have more money to improve the quality under Case III than under Case II.

Proposition 7. under certain conditions, the network externalities and software compatibility can change the impact of open source software on the optimal functional quality of proprietary software.

Proof. (i) setting $y = [v_{po} - \alpha(1 - k)][4r(v_p - \alpha) - 1] - v_p\{4r[v_{po} - 2\alpha(1 - k)] - 1\} - f_o[4r(v_p - \alpha) - 1]$ and then rearranging it, we can obtain $y = 4r(1 - k)\alpha^2 + [-4r(v_{po} - f_o) + (4rv_p + 1)(1 - k)]\alpha + [(v_{po} - f_o)(4rv_p - 1) - v_p(4rv_{po} - 1)]$. When $k = 0$, $y = 4r\alpha^2 + [4r(v_o + f_o) + 1]\alpha + v_o + f_o - 4rv_pf_o$. We set $a = 4r$, $b = 4r(v_o + f_o) + 1$ and $c = v_o + f_o - 4rv_pf_o$. When $r > \frac{v_o + f_o}{4v_pf_o}$, it exists $\alpha^* = \frac{-b+\sqrt{b^2-4ac}}{2a} > 0$ meeting that $y < 0$ if $0 < \alpha < \alpha^*$ and $y > 0$ if $\alpha > \alpha^*$. Combing with Proposition 6, $f_o > \xi$ and then $f_p^{\&} > f_p^*$ when $y < 0$; $f_o < \xi$ and then $f_p^{\&} < f_p^*$ when $y > 0$; (ii) rearranging y, we can obtain $y = \alpha[4r(v_p + \alpha) + 1](1 - k) + v_o + f_o - 4r[v_{po}\alpha + (v_p - \alpha)f_o]$. We set $a' = \alpha[4r(v_p + \alpha) + 1]$ and $b' = v_o + f_o - 4r[v_{po}\alpha + (v_p - \alpha)f_o]$. If $\alpha > 0$, $r > \frac{v_o + f_o}{4[v_{po}\alpha + (v_p - \alpha)f_o]}$ and $0 < \frac{4r[v_{po}\alpha + (v_p - \alpha)f_o] - (v_o + f_o)}{\alpha[4r(v_p + \alpha) + 1]} < 1$, it exists $k^* = 1 + \frac{b'}{a'} > 0$ meeting that $y < 0$ if $k^* < k \leq 1$

and $y > 0$ if $0 \leq k < k^*$. Combing with Proposition 6, $f_o > \xi$ and then $f_p^{\&} > f_p^*$ when $y < 0$; $f_o < \xi$ and then $f_p^{\&} < f_p^*$ when $y > 0$.

The above proposition shows that, comparing the optimal results for case II and case III, the presence of open source software can cause the functional quality of proprietary software increase under some network externalities intensities (resp. compatible degrees), while it can cause the functional quality of proprietary software decrease under others. Table 1 and Table 2 give specific examples: given $f_o = 0.1$, $v_o = 1$, $v_p = 2$, $k = 0$ and $r = 2$, $f_p^{\&} > f_p^*$ when $\alpha = 0.01$, and $f_p^{\&} < f_p^*$ when $\alpha = 0.1$; given $f_o = 0.05$, $v_o = 1$, $v_p = 2$, $\alpha = 0.1$ and $r = 3$, $f_p^{\&} > f_p^*$ when $k = 1$, and $f_p^{\&} < f_p^*$ when $k = 0$.

	$f_p^{\&}$	f_p^*
$\alpha = 0.01$	0.1340	0.1301
$\alpha = 0.1$	0.1408	0.1481

Table 1. The impact of network externalities on optimal qualities under case II and case III
(when $f_o = 0.1$, $v_o = 1$, $v_p = 2$, $k = 0$ and $r = 2$)

	$f_p^{\&}$	f_p^*
$k = 0$	0.0917	0.0988
$k = 1$	0.0917	0.0864

Table 2. The impact of compatibility on optimal qualities under case II and case III
(when $f_o = 0.05$, $v_o = 1$, $v_p = 2$, $\alpha = 0.1$ and $r = 3$)

Since the network externalities and compatibility may change how open source software affects the quality of proprietary software, we must consider this effect when investigate the quality choices for proprietary vendors. However, most of the relevant research does not consider the network externalities and software compatibility between proprietary and open source software (Raghunathan et al., 2005; Choudhary & Zhou, 2007; Xing, 2012).

5. Conclusions

Through modifying the vertical differentiation model, this paper analyzes the impact of competition from open source software on the quality of proprietary software when the market exhibits positive network externalities. The work assumes that the usability of open source software is inferior to proprietary software and mainly finds that: (i) the impact of network externalities on the functional quality of proprietary software may depend on the software

compatibility and the quality of open source software; (ii) the compatibility between proprietary and open source software may affect the quality choices of proprietary vendor; (iii) the impact of the functional quality and usability of open source software on the functional quality of proprietary software may differ under certain conditions; (iv) compared with the case of proprietary software competing with open source software, the optimal functional quality of proprietary software is higher under the case of proprietary software monopolizing the whole market, while it may be lower under the case of proprietary software monopolizing only part of the market. That is, the appearance of open source software may lead to the decrease (or increase) of proprietary software's functional quality; (v) under certain conditions, the network externalities and software compatibility can change the impact of open source software on the functional quality of proprietary software. Nevertheless, no one method is perfect. The study will be interesting if the propositions given in the paper are empirically validated through example cases. An empirical approach to validating the propositions will be provided in future research.

Acknowledgements

The author would like to thank the financial support from Natural Science Foundation of Shandong Province (No. ZR2013GL005), Social Science Planning Research Project of Shandong Province (No. 12CJRJ17), Shandong province soft science research plan (No. 2014RKB01165) and Shandong Higher School of Humanities and Social Science Research Projects (No. J13WF11).

References

Chen, H.K., Liu, Y., & Tang, Q. (2011). The impact of network externalities on the competition between open source and proprietary software. *Journal of Management Information Systems,* 27(4), 201-230. http://dx.doi.org/10.2753/MIS0742-1222270407

Choudhary, V., & Zhou, Z.Z. (2007). Impact of competition from open source software on proprietary software. *Working Paper Series,* No. 08070. China Economics and Management Academy. Available at: http://cema.cufe.edu.cn/admin/data/uploadfile/200810/20081006225900419.pdf

CIO (2002). *Open source gains momentum*.

Dalle, J., & Jullien, N. (2002). Open-source vs. proprietary software. *Working paper.* Available at: http://opensource.mit.edu/papers/dalle2.pdf

Gramstad, A.R. (2014). *Piracy in commercial vs. open-source software competition.* Available at: http://www.sv.uio.no/econ/english/research/news-and-events/events/conferences/2014/papers/gramstad_norio15may.pdf

Katz, M., & Shapiro, C. (1985). Network externalities, competition, and compatibility. *American Economic Review,* 75, 424-440.

Lanzi, D. (2009). Competition and open source with perfect software compatibility. *Information Economics and policy,* 21(3), 192-200. http://dx.doi.org/10.1016/j.infoecopol.2008.11.004

Lin, L.H. (2008). Impact of user skills and network effects on the competition between open source and proprietary software. *Electronic Commerce Research and Applications,* 7(1), 68-81. http://dx.doi.org/10.1016/j.elerap.2007.01.003

Mussa, M., & Rosen, H. (1978). Monopoly and Product Quality. *Journal of Economic Theory,* 18, 301-317. http://dx.doi.org/10.1016/0022-0531(78)90085-6

Meng, Z., & Lee, S.Y. (2005). *Open source vs. proprietary software: competition and compatibility.* Paper provided by EconWPA in its series industrial organization with number 0508008. Available at: http://ssrn.com/abstract=780804

O'Reilly, T. (1999). Lessons from open-source software development. *Communications of the ACM,* 42(2), 33-37. http://dx.doi.org/10.1145/299157.299164

Pradniwat, P. (2008). Competition strategy between proprietary and open source software in the presence of software piracy and network externalities. *Thammasat Economic Journal,* 26(1), 128-170.

Raghunathan, S., Prasad, A., Mishra, B.K., & Chang, H. (2005). Open source versus closed source: software quality in monopoly and competitive markets. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans,* 35(6), 903-918. http://dx.doi.org/10.1109/TSMCA.2005.853493

Shy, O. (2001). *The economics of network industries.* Cambridge University Press. http://dx.doi.org/10.1017/cbo9780511754401

Wheeler, D.A. (2007). *Why open source software/free software (OSS/FS, FLOSS or FOSS)? Look at the numbers!* Available at: www.dwheeler.com/oss_fs_why.html

Xing, M.Q. (2010). The quantity competition between open source and proprietary software. *Proceedings of International Conference on Information Management, Innovation Management and Industrial Engineering.* 184-187. http://dx.doi.org/10.1109/iciii.2010.50

Xing, M.Q. (2012). Impact of open source software on the quality of proprietary software and software differentiation. *Journal of Convergence Information Technology,* 7(20), 242-249. http://dx.doi.org/10.4156/jcit.vol7.issue20.29

Xing, M.Q. (2013). Competition between free open source, commercial open source and proprietary software. *Journal of Communications,* 8(10), 665-671. http://dx.doi.org/10.12720/jcm.8.10.665-671

Xing, M.Q. (2014a). The impact of commercial open source software on proprietary software producers and social welfare. *Journal of Industrial Engineering and Management,* 7(5), 1183-1196. http://dx.doi.org/10.3926/jiem.1260

Xing, M.Q. (2014b). On the optimal choices of R&D risk in a market with network externalities. *Economic Modelling,* 38, 71-74. http://dx.doi.org/10.1016/j.econmod.2013.12.024

Zeroukhi, M., & Pénard, T. (2014). Open source software subsidies and network compatibility in a mixed duopoly. *Economics Bulletin,* 34(2), 1174-1184.

Journal of Industrial Engineering and Management, 2015 (www.jiem.org)