University of Navarra

Working Paper

WP No 519

September, 2003

DYNAMIC MIXED DUOPOLY: A MODEL MOTIVATED BY LINUX VS. WINDOWS

Ramón Casadesus-Masanell *
Pankaj Ghemawat**

* Professor of General Management, IESE

** Professor, Harvard Business School

IESE Business School - University of Navarra
Av. Pearson, 21 - 08034 Barcelona, Spain. Tel.: +34 93 253 42 00 Fax: +34 93 253 43 43
Camino del Cerro del Águila, 3 (Ctra. de Castilla, km. 5,180) - 28023 Madrid, Spain. Tel.: +34 91 357 08 09 Fax: +34 91 357 29 13

DYNAMIC MIXED DUOPOLY:
A MODEL MOTIVATED BY LINUX VS. WINDOWS

Abstract

This paper analyzes a dynamic mixed duopoly in which a profit-maximizing competitor interacts with a competitor that prices at zero (or marginal cost), with the cumulation of output affecting their relative positions over time. The modeling effort is motivated by interactions between Linux, an open-source operating system, and Microsoft's Windows in the computer server segment, and consequently emphasizes demand-side learning effects that generate dynamic scale economies (or network externalities). Analytical characterizations of the equilibrium under such conditions are offered, and some comparative static and welfare effects are examined.

Keywords: Open-source software, Network effects, Microsoft, Linux, Competitive dynamics, Strategy.

DYNAMIC MIXED DUOPOLY:
A MODEL MOTIVATED BY LINUX VS. WINDOWS

I. Introdution

Mixed duopoly refers, in this paper, to the interactions between a not-for-profit competitor and a for-profit competitor. While this formalization is motivated by the case of Linux vs. Windows that is described below, asymmetry with respect to profit maximization also seems focal in the sense of representing the most obvious modification to the standard assumption of symmetric profit maximization. Specifically, it is assumed that the not-for-profit player prices its product at zero (or at marginal cost) and that the for-profit player must take that commitment as given in making its own pricing decisions. The stylization evokes not only interactions between open-source software development efforts (of which Linux is one of many) and their for-profit competitors (of which Microsoft is one of many) but aspects of a number of other types of interactions as well. These include interactions between a profit-maximizer and a competitor pursuing volume or market share by pricing at marginal cost, between profit-maximizers and much more patient competitors, between private and state-owned/supported enterprises (e.g., Boeing vs. Airbus, in the official U.S. view), between for-profit firms and nonprofits, or even the social sector, broadly defined (e.g., between pharmaceutical firms and universities in the life sciences—although those relationships involve complementarities and side-payments as well as somewhat fragmented competition), and between practice and research more generally (e.g., between management consulting firms and business schools—which also involve cooperation and competition in fragmented settings). As a first cut at pushing forward from the present state of analysis of mixed duopolies, this paper focuses on the case in which the relationships between the two players are basically competitive (e.g., as in Linux vs. Windows), but bringing cooperative relationships into the picture would be an obvious and important next step.

Previous theoretical analyses of mixed duopolies, as defined above, have mostly been static. The prototypical model of this sort analyzes competition between a profit-maximizer and a sales-maximizer and confirms that if their products are substitutes, the profit-maximizer fares worse, in terms of both volume and prices, than it would if it were facing another profit-maximizer (as in the standard set-up). Specifically, if the sales-maximizer prices at zero or at a common marginal cost, the profit-maximizer can make positive operating profits only to the extent that the two's products are imperfect substitutes for each other. Especially since the sales-maximizer's volume is predicted, *ceteris paribus*, to be higher than the profit-maximizer's, might the former somehow displace or push out the latter from the market? That and a number of other questions (e.g., welfare implications, comparative statics) animate the explicitly dynamic model of mixed duopoly presented in this paper.

Note: We thank Miguel Angel García Cestona, Fabio Maneti, Patrick Moreton, and seminar participants at NYU, the 2003 Strategy Research Forum held in Washington University, St. Louis, the 1st Meetings of the International Industrial Organization Society, held in Boston, and the 2003 Society for the Advancement of Economic Theory (SAET) conference, held in Rhodes, for useful comments.

There are a number of interesting ways in which one might add some dynamics to the prototypical, static model of mixed duopoly. The extensive literature on competition in the presence of learning-by-doing initiated by Spence [1981] seems particularly pertinent since it focuses directly on how cumulated output can reduce costs (or, less commonly, improve willingness-to-pay) over time. The present paper can be read as an attempt to extend that literature to a duopoly structure in which objectives are mixed rather than symmetric. The focus falls on demand-side learning here, both because of the specifics of the Linux vs. Windows case and because demand-side learning is entirely unstudied from a mixed perspective (whereas there *are* a few models of cost-side learning that can at least partially be reinterpreted in these terms, e.g., models in which a dominant firm that is a price leader competes with a price-taking fringe, e.g., Ross [1986]).

The focus on modeling the effects of learning on the demand side induces some simplifications on the cost side. The baseline model in this paper assumes that marginal costs are zero for both competitors. Cases with symmetric constant marginal costs are structurally equivalent, and an extension to allow for a marginal cost penalty for the profit-maximizing competitor is offered later on in the paper. Some of the conclusions can also be reinterpreted to apply to cases in which the profit-maximizer incurs fixed cost flows for as long as it chooses to remain in operation as well as marginal cost flows. However, the full endogenization of fixed cost investments in learning, as opposed to learning purely by doing, is beyond the scope of this paper because of complexities that are hard to handle even within the familiar confines of symmetric profit-maximization.

The ultimate test of all these design choices that are embedded in the model of dynamic mixed duopoly developed later in this paper is whether they lead anywhere interesting. Several criteria, in addition to the basic requirement of theoretical coherence, might be specified for such assessments: nonobviousness, breadth of applicability, policy implications, et cetera. But the particulars of the model must be specified and the implications identified before assessments of whether they are interesting can be made. To this end, Section II provides some background on Linux (and open-source software more generally) vs. Windows, the case in which the model is grounded in a number of ways. Section III lays out the basic model and comparative statics, and Section IV develops some extensions. Section V concludes.

II. Background: Linux, Open Source Software and Microsoft

The analyses of mixed duopoly in Sections IV and V of this paper are grounded in the case of Linux vs. Windows in operating system software for computer servers that is discussed in some detail in this section (and even more extensively in a teaching case). The detail reflects the multiple ways in which such grounding helps with the modeling effort: by helping determine some of the assumptions underlying the modeling effort, suggesting some of the analyses to be performed with it, providing a basis for testing a (limited) number of its implications as well as illustrating them, and also supplying some concrete, vivid ways of representing the players in the model, their interactions, et cetera. (Linux vs. Windows is more memorable and less likely to lead to confusion about objectives and roles than firm 1 vs. firm 2, for example.)

Despite this last use (or abuse) of the case, the model should *not* be thought of as a literal representation of Linux vs. Windows: the references to the two in the later, analytical sections are mostly meant to be metaphorical and *not* to suggest that a serious or deep policy

analysis of that particular case is being offered[1]. As discussed in Section I, the objectives of the modeling effort are actually broader than developing a model of open-source software vs. traditional, for-profit software firms, although the broader issues are apparent even within that particular setting. Specifically, there has been an extensive, and continuing, debate about open-source as a system for innovation that ranges, at its extremes, from those who celebrate open source in frankly liberationist terms to those who condemn it as an innovation destroyer ("a threat to the American way," in the words of one senior Microsoft executive).

In 2003, the Linux operating system was the best-known example of the burgeoning open-source software movement[2]. Other major open-source successes included Apache, the most popular software for running web servers, Sendmail, the dominant messaging service program for routing and handling email used by email servers, and PERL, a programming language for writing scripts allowing websites to call and run applications on a server. Open-source's smaller successes in enabling users to develop their own software applications were considered no less important, however, in its campaign to open up access to computer software "source code".

Open-source software development stressed collaboration among user-developers, with each being able to directly alter and improve the product. Traditionally, when programmers modified code, they altered "source code" written in high-level computer languages such as Java, C++, and Unix that was compiled and then read by machines as "object code," expressed in machine language (a string of 0s and 1s) that was hard for humans to interpret. Open-source projects let users directly modify source code, which was freely available. This was supposed to lead to continuous improvement of the product, as bugs were patched and new capabilities were developed. Traditional or "closed" software, on the other hand, required that consumers purchase a license to run object code rather than gaining access to source code. Open-source software was distributed with "its sources and the right to modify and redistribute it…" intact such that the original development paradigm would continue. Open-source thus denoted itself not only a particular method for the creation of software but also its distribution under a particular institutional framework.

The institutional framework for making a public good of software was provided by the GNU Public License, or GPL, a new type of copyright created by the Free Software Foundation in the late 1980s. The GPL prohibited developers from making code proprietary, by preserving the right of anyone to "use it, copy it, modify it, and distribute their modifications." Under GPL, derived works also had to be distributed under the same format, which had important implications. This caveat prevented "software hoarding," or taking code

---

[1] For instance, in early 2003, the migration rate to Linux seemed highest among server customers previously using Sun's proprietary stack of hardware and software with its own operating system variant on Unix, Solaris. While migration of this sort could be reinterpreted in terms of the duopoly model developed and analyzed later in this paper as restricting Windows' share growth, really taking it seriously would require a somewhat different, more complex model—a triopoly at least.

[2] In this paper we use the expression "open source" to refer to both open source and free software (as defined by Richard Stallman's Free Software Foundation (FSF)). According to the FSF, a program is free software if users have four fundamental freedoms: freedom to run the program for any purpose, freedom to study how the program works and adapt it to the user's needs, freedom to redistribute copies, and freedom to improve the program and release improved versions to the public. This view is shared by the open source movement. However, according to Richard Stallman, "the fundamental difference between the two movements [open source and free software] is in their values, their ways of looking at the world. For the Open Source movement, the issue of whether software should be open source is a practical question, not an ethical one. As one person put it, 'Open source is a development methodology; free software is a social movement.' For the Open Source movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution."

"private." Lastly, in order to avoid the "closing" of code, the GPL also prohibited mixing open-source with closed source code at the "source code" level. In economic terms, the GPL license attempted to establish the "public good" character of the software that it covered. This copyright form quickly became known as "copyleft," as it was diametrically opposed to traditional copyright forms. The copyleft license was maintained by the Free Software Foundation, and enforced through Internet verification, as users clicked an "I agree" button when downloading copyleft software. The copyleft license was offered on take-it-or-leave-it terms and without it, software was not considered open-source. The provisions of the GPL had yet to be tested in court, however.

Open-source software development occurred within the GPL framework and involved efforts that varied greatly in terms of the scale and organizational complexity but generally sought to harness demand-side learning more effectively than traditional "closed" models. Open-source development was supposed to compress development cycles, lead to more "use-combinations" being tested and provide more of an incentive for users to report problems or fixes than "closed" models, which might make users pay for improvements even if they had suggested them. Successful open-source software applications such as Apache and Linux did, on a number of dimensions (e.g., defect rates and speed of response to customer problems), exhibit higher quality than their leading for-profit competitors (Microsoft's IIS and Windows NT respectively)—evidence concerning Linux in this regard is discussed in a bit more detail below.

Such quality-enhancing, demand-side learning effects had been compared to conventional cost-reducing supply-side learning curves with their traditional industrial logic of cutting price, gaining share and reducing costs particularly rapidly (e.g., by the Boston Consulting Group, an early proponent of the strategic importance of supply-side learning-by-doing). In the context of open source software, the virtuous cycle was supposed to involve giving source code away, attracting users through performance advantages as well as zero prices, and drawing on users' learning and contributions to increase product quality particularly rapidly. These efforts to harness demand-side learning had a number of essential elements. First, they presumed a large number of users to report bugs—as evident in the aphorism that "Given enough eyeballs, all bugs are shallow." Second, smaller numbers of users performed such essential functions as support, documentation and debugging and a smaller number still developed most of the new code—although here too, there was need for critical mass. Obviously, it helped if the users who valued the open source product more also had the programming ability to customize it to their purposes. As a result, engineering tools and utilities made up a large percentage of open-source programs while word processors did not. Third, good ideas—including all the major open-source successes cited above—were generally thought to come from developers "scratching an itch" and providing their services for free instead of following orders or trying to make money. Fourth, the leaders of many initiatives adopted policies of "release early, release often"—an emphasis greatly aided by the Internet, which permitted people around the world to work on tasks such as software development in stable, coordinated ways. Fifth, to the extent that the product was complex, it had to be disaggregable into parallel modules for open-source development to work well. Finally, open-source projects often complemented each other: for example, GNOME, also an open-source project, was creating a graphical user interface (GUI) for Linux. Open-source projects additionally helped each other in the sense that when one project gained legitimacy, that created a halo-effect for the entire stable of open-source programs.

Linux was the most visible of myriad open-source projects: it was cited much more often than any other project, more often even than the open-source movement itself. Linux was an operating system that reflected the contributions of over 3,000 developers in ninety

countries and five continents. Operating systems performed basic, relatively discrete and well-defined tasks, such as recognizing input from the keyboard, sending output to the display screen, keeping track of files and directories on the disk, and controlling peripheral devices such as disk drives and printers. At the heart of any operating system was the kernel which, in Linux's case, made up about 30% of the total number of lines of code in the OS. Surrounding the kernel were modules: applications, utilities, ports etc.

The Linux initiative began in 1991, when Linus Torvalds, then a young undergraduate at the University of Helsinki, began to develop Linux as a "Unix-like" kernel. By October 1991, he released the first version of his "pet project" and by December, over one hundred people had joined the Linux newsgroup mailing list. In 1992, Torvalds integrated his work with that already completed on another open-source project, GNU, to create a freely-available Unix-like OS. Development then began in earnest, although it was initially subject to significant hardware constraints. The size of the Linux kernel grew nearly exponentially, from approximately 1 Megabyte (Mb), compressed, by the end of 1992 to 5 Mb by 1996 to about 20 Mb by 2000. This growth brought both new functionalities and new problems, as complexity grew. By July 2000, there were over four hundred Linux usergroups, over 3,500 applications and more than 85,000 Linux-related messages generated per month. And Linux had carved out a significant #2 position for itself in server OSs (for the powerful computers that were the backbones of networked communicating), where it was narrowing the market share gap with the dominant offering from Microsoft, but continued to have very little presence in client OSs (for personal computers).

Server OSs were bought almost exclusively by corporations and the market for them was segmented based on whether they were supposed to operate high-end servers that cost more than $1 million, mid-range servers that cost from $100,000 - $1 million and the entry-level segment that typically consisted of servers costing less than $100,000. Within the server OS market, Linux's penetration was concentrated in the entry-level segment.

According to one set of estimates, the total number of Linux users was estimated to have increased from 1,000 in 1992 to 20 million by 2000. The same source predicted that by 2004, the number of Linux users would increase to 30 million, across both client and server OSs, compared to 338 million for Microsoft products. Furthermore, by 2004, Linux was expected to secure a 6% share of the client OS market and 37% of the server OS market of installed users, compared to 88% and 43%, respectively, for Microsoft's market-leading products. Focusing on the server OS market, Linux accounted for 2.0 million new licenses worldwide in 2000, a 30% unit share, while Microsoft had 2.5 million new licenses of Windows NT during the same year, a 38% unit share. Through 2004, the compounded annual growth rate in unit server OS license shipments was forecast to be 35% for Linux, versus somewhere between 15% and 25% for Microsoft. Table 1 provides additional market share data.

Table 1. Market Share for Server Operating Systems (in percentage)

System	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003
Microsoft	7.0	18.1	25.6	35.3	38.3	38.1	38.5	39.5	40.5	41.0
Novell	39.6	34.7	32.1	26.7	22.8	19.1	15.0	13.0	12.0	10.0
Linux	0.0	0.0	6.5	6.8	15.8	24.8	30.0	34.0	36.0	38.0
Unix	28.6	25.4	20.1	20.9	18.8	15.5	15.0	13.0	12.0	10.0
Other	11.0	8.0	4.5	3.9	1.3	1.0	5.0	3.0	2.0	2.0

Sources: IDC, Brian Silverman, "Sun Microsystems, Inc.: Solaris Strategy", HBS (February 2, 2001): 9-701-058.

Supporters explained Linux's success at the expense of Microsoft's Windows and other closed server OSs in terms of numerous advantages: lower costs (see Table 2), fewer bugs and more reliability, which were related to each other and to faster releases, better interoperability across different computer platforms (since developers of new platforms could alter Linux themselves), greater scalability (because it was bare-bone rather than feature-heavy), modularity (since even the kernel was split into kernel modules starting with Linux version 2.0), freedom from restrictions on the number of users attached to a server or the number of servers on which a single copy of the OS could be installed, and the convenience of being able to download the software from the Internet instead of having to obtain it in a shrink-wrapped package. Microsoft disputed many of these advantages, but the fact that there was a dispute at all—compared to earlier, when Microsoft took no public notice of Linux—suggested that this open-source development effort, like several others, had reached a level of acceptance that Microsoft regarded as threatening.

Table 2. Linux vs. Microsoft Costs: Typical File/Print Server Network

Baseline:	**Linux**	**Microsoft**
Users supported	1,000	1,000
Number of servers required	1	4
Software:		
OS license per server	$99	$4000
Client license	$0	$128
Total software cost	$99	$144,000
Hardware:		
Cost per server	$6,000	$6,000
Installation/server	250	250
Total hardware cost	6,250	25,000
Integration:		
Time per server	32	16
Cost per hour	250	125
Total integration costs	8,000	8,000
Total Cost:	14,349	177,000

Source: Boston Consulting Group.

Microsoft appeared to stand to lose more than any other company from Linux's penetration of the market and, more generally, open-source software's success, precisely because of how well it had performed in the past: it was number 1 or 2 in all packaged software categories, its flagship Windows product was installed on 95% of desktops, and it had delivered a 58% average annual return to investors from 1989-1999. However, up to 25% of Microsoft's revenue was estimated to be at significant risk once threats from open-source software were taken into account (see Table 3). In addition, market dominance did not translate into resource dominance: Microsoft had had only 250 people working on Windows NT—whose costs it had to shoulder in entirety—compared to the thousands of developers working on Linux, mostly for free. In fact, some thought that Microsoft's dominance was one of the biggest spurs to the investment in and success of open-source software. In addition,

allegations of market dominance by Microsoft had also engendered significant antitrust concerns and constraints.

Table 3. Open-source Threats to Microsoft

Products	2000 revenue ($M)	Percentage of Total	Risk Level	Open Source attackers
Server OS				
NT Server	1,377	6	High	Linux, Apache, BSD, Gnome
Windows 2000	1,836	8	High	Gnome, KDE, SendMail, *BSD, PHP Perl
Client OS				
Windows 98/ME	4,56	20	Medium	Linux, GNOME, Helix, Eazel
NT Workstation	1,607	7	High	Linux, GNOME, KDE
Applications				
MS Office	8,611	38	Low	Open Office, Gnumerics, K-Office
Exchange Server	942	4	Medium	Sendmail
SQL Server	914	4	High	MySQL, PostGRES, InPRISE, SAP
Internet Explorer	791	3	Medium	Mozilla
Others				
MSN	2,089	9	NA	
Misc	229	1	NA	
TOTAL	22,956	100		

Source: Tucker Anthony Capital Markets

One of Microsoft's first acknowledgements of Linux as a viable competitor was in 1998, in documents later leaked (http://www.opensource.org/halloween), in which Microsoft employees laid out plans for defending Microsoft's franchise by moving to decommoditize protocols and services by extending them in a proprietary manner," thereby "locking out customers and competitors." In May 1999, Microsoft assigned a ten person "swat team" to work specifically on the problem of Linux. And in July 2000, Microsoft announced its "dot-net initiative" which sought to sell more software products as services (an application service provider or ASP model) rather than as shrink-wrapped packages. The ASP model would afford Microsoft more flexibility to try different pricing schemes, including a low-priced subscription model that might compete effectively with open-source software.

Since then, Microsoft had stepped up the intensity of its rhetoric. Microsoft CEO Steven Ballmer characterized Linux as "enemy no. 1." Another senior executive described open-source as the "worst thing to happen to the software industry" and the "thing that kills innovation," and painted a bleak picture of a world in which immature products were released and consumers suffered. While criticizing the GPL, Microsoft talked up its own "Shared Source Philosophy (SSP)," which claimed to mine the best aspects of open-source and which would be incorporated into Microsoft's licensing model. Elements of the SSP included open-source access programs (some already mandated by the courts), educational initiatives, and measures to improve customer feedback. In the words of a senior Microsoft executive, SSP was a "balanced approach that allows us to share source code… while maintaining the intellectual property."

Even more recently, in summer 2002, Orlando Ayala, then in charge of worldwide sales at Microsoft, sent a confidential e-mail message (later leaked) to senior managers, including CEO Ballmer, telling executives that if a deal involving governments or large institutions hung in the balance, they could draw on a special fund to offer the software at a

steep discount or even free if necessary. "Under NO circumstances lose against Linux," according to the email[3].

Such interactions between not-for-profit open-source software and for-profit closed software have not been studied much in the burgeoning literature on open source software, which mostly focuses on how open-source development efforts are organized, particularly the satisfaction of the individual participation constraints of user-developers who are critical to learning on the demand side (see Appendix A). In particular, there seems to be a total dearth of models that embed the competition between Linux and Windows in an explicitly dynamic model with demand-side learning. Such a model is presented in the next section.

III. A Model of Open Source vs. Closed Software

The baseline model that we work with assumes a mixed duopoly, demand-side learning for both open-source and closed development efforts, a commitment to price the open-source product at zero, and strategic but nondiscriminatory pricing by the for-profit player that accounts for the effects of current prices on learning and future customer appeal.

We begin by specifying the demand side of the model. In each period *t*, a new cohort of potential users enters the market. We normalize the size of this cohort to 1. Let $y_i(t) \in \mathbf{R}_+, i \in \{W, L\}$ be the cumulative market share of operating system (OS) *i* at time t. Thus, if q(τ) is the portion of individuals in time τ's cohort who buy Windows, then

$$y_W(t) = \int_0^t q(\tau)\, d\tau \quad \text{and} \quad y_L(t) = \int_0^t (1 - q(\tau))\, d\tau$$

where we use *W* and *L* to denote Windows and Linux, respectively. Thus, we assume that every individual in each cohort uses one and only one OS; she either buys Windows or downloads Linux for free.

Let $y(t) \equiv y_W(t) - s y_L(t)$ where *s* is a scalar greater than zero. Let α_i denote OS *i*'s value by the cohort entering at time *t* and let $\alpha_i = \alpha_i(y(t))$. We refer to $\alpha_i(y(t))$ as OS *i*'s technological trajectory (Foster, 1988). OS *i*'s trajectory is a function of its cumulative market share, $y_i(t)$, and the competing OS's cumulative market share, $y_{-i}(t)$. While technological trajectories are exogenously given in the model, how far each OS can travel down its trajectory is endogenous; the result of the dynamics of competition.

We assume that:

1. We assume linear demand functions: Windows' value to customer $q \in [0,1]$ is

$$\alpha_W(y)(1-q) \tag{1}$$

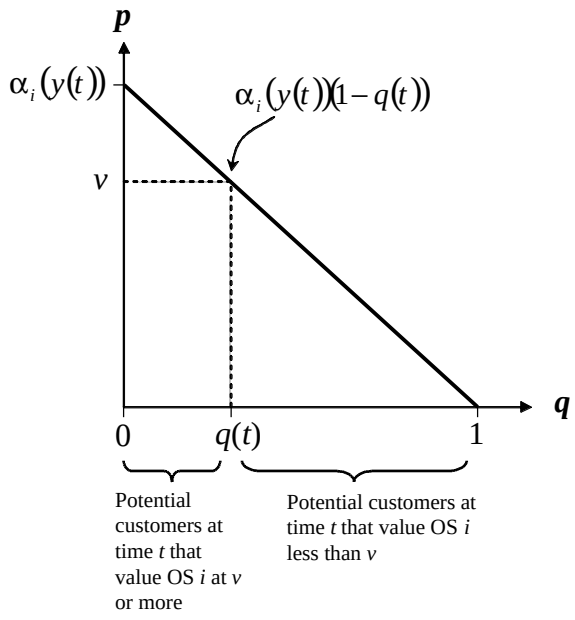
Similarly, let the value of Linux be

$$\alpha_L(y)(1-q) \tag{2}$$

---

[3] Thomas Fuller, "How Microsoft Warded off Rival," *New York Times*, May 15, 2003.

Graphically,



2. $\frac{\partial \alpha_i(y)}{\partial y_i} > 0$, i.e., that OS *i*'s value increases with OS *i*'s cumulative market share. This captures two kinds of effects: the more people use (or have used) a given OS, the more feedback is likely to have been provided for improvement. In the case of open source projects, users can make improvements directly on the code. In the case of closed software, users can call up or email the software developer with suggestions. In addition, the larger $y_i(t)$ is, the more complements are likely to be available for OS *i*. Availability of complements increases the value of the OS.

The assumption also implies that $\frac{\partial \alpha_i(y)}{\partial y_{-i}} < 0$, i.e., OS *i*'s value decreases as the cumulative market share of the competing OS increases. Again, this is related to the attention and effort that third-party developers devote to creating new and improving old software and hardware. An OS is more likely to get developers' attention if its cumulative market share is relatively large. The larger $y_{-i}(t)$ (holding $y_i(t)$ constant), the less effort will be devoted to developing complements for OS *i*, and vice versa. This reduction jeopardizes the value of OS *i* because bugs are not fixed as often, programs are not updated, new software and hardware may not work/communicate as well with existing software and hardware, unforeseen compatibility issues are more likely to arise, et cetera.

3. $\lim_{y \to \infty} \alpha_i(y) = \bar{\alpha}_i < \infty$, i.e., OS *i*'s value is finite, even if everyone uses the OS. Also, $\lim_{y \to -\infty} \alpha_i(y) = 0$, i.e., OS *i*'s value approaches zero if everyone uses the other OS. Thus, technological trajectories are bounded; there is an upper bound on the maximum value created by each operating system. The case in which $\bar{\alpha}_L > \bar{\alpha}_W$ corresponds to a situation where Linux's potential quality is strictly larger than that of Windows. Assumptions 2 and 3 imply that $\alpha_i(y) \geq 0$, i.e., the value of each OS is always positive.

4. Let $\beta(y) \equiv \alpha_W(y) - \alpha_L(y)$, then $\frac{d^2\beta(y)}{dy^2} \leq 0$ for $y>y^0$, where y^0 is the value of y for which both Linux and Windows are perceived as equally valuable. (Formally, y^0 solves $\alpha_W(y) = \alpha_L(y)$; assumptions 2 and 3 imply that y^0 is unique.) This assumption says that cumulative market share has a decreasing marginal effect on the difference in value between Windows and Linux[5].

As mentioned above, we assume that, everything else constant, as y_i grows, the vertical intercept $\alpha_i(y)$ in the demand function also increases. Thus, as the accumulated market share (or installed base) of OS *i* grows, the value of the OS also grows for *all* potential customers.

Recall that $y(t) \equiv y_W(t) - sy_L(t)$. Formally, *s* is the absolute value of the derivative of *y* with respect to y_L. If $s>1(<1)$, then increases in y_L have more (less) of a positive impact on perceived quality of Linux than the negative impact of comparable increases in y_W. Parameter *s* has two complementary interpretations: the differential in demand-side learning between Linux and Windows and the differential strength of network externalities due to the availability of complementary software. Thus, for a given level of network externalities due to complements, increases in *s* correspond to a strengthening in Linux's demand-side learning.

Example. The following S-shaped technological trajectories satisfy the assumptions[6].

$$\alpha_W(y) = \begin{cases} \bar{\alpha}_W - \frac{\lambda}{1+y} & \text{if } y_W \geq sy_L \\ \frac{\bar{\alpha}_W - \lambda}{1-y} & \text{otherwise} \end{cases}$$

Let $\lambda \in (0,1)$. Let

$$\alpha_L(y) = \begin{cases} \frac{\bar{\alpha}_L - \lambda}{1+y} & \text{if } y_W \geq sy_L \\ \bar{\alpha}_L - \frac{\lambda}{1-y} & \text{otherwise} \end{cases}$$

and

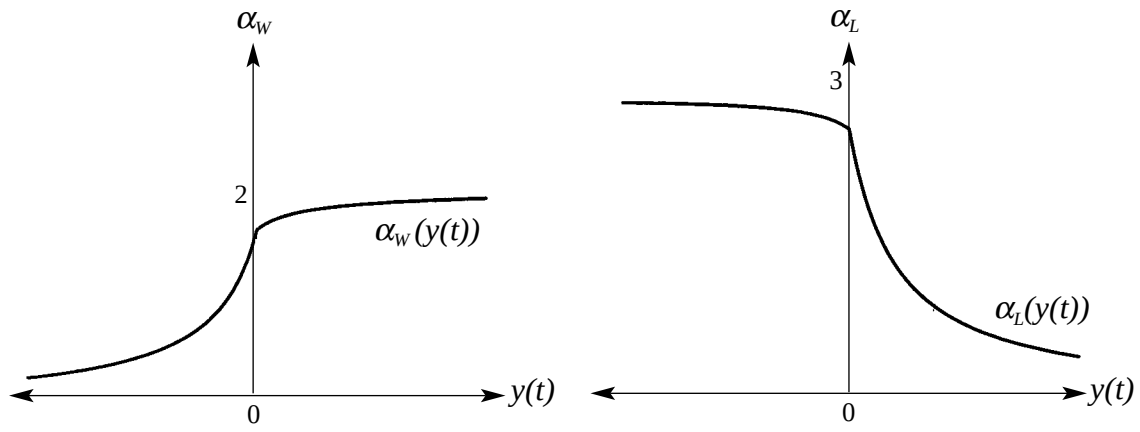
In this example, Windows is more valuable than Linux whenever $y_W > \frac{\bar{\alpha}_L - \bar{\alpha}_W}{\bar{\alpha}_W} + sy_L$.

Assuming $\lambda = \frac{1}{3}$, $\bar{\alpha}_W =$ 2 and $\bar{\alpha}_L =$ 3, then functions α_W and α_L look as follows:

---

5 This is a technical assumption that simplifies exposition. It can be relaxed to: there is $\hat{y} \geq y^0$ such that for all $y \geq \hat{y}$, $\frac{d\beta(y)}{dy} < 0$. With this assumption, there may be more than two steady states. If there are multiple steady states, our results on y^{SS} (the stable steady state) hold unchanged for the steady state with the largest *y*.

6 Strictly speaking, the functional forms in the example satisfy the assumptions at all points other than $y = 0$, because at this point the trajectories are not differentiable.



In what follows, we analyze the dynamics of competition and its effects on cumulative market shares y_W and y_L.

Dynamics of competition

We distinguish between two cases: one in which Microsoft is a monopolist and another in which it is a duopolist, competing to sell Windows against Linux.

Monopoly

In a monopolistic market structure, there is no substitute for Windows and potential customers are willing to pay something (even only a small amount) for Windows. Inverse demand follows directly from equation (1). Let r be Microsoft's discount rate and assume that marginal cost of an extra copy of Windows is zero. Microsoft solves:

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t)\, p(t)\, dt$$

subject to

$$\dot{y}_W = q(t)$$
$$p(t) = \alpha_W(y(t))(1 - q(t))$$
$$p(t) \geq 0$$

The following proposition will be useful in comparing monopoly and duopoly.

Proposition 1 *As* $t \to \infty$, $p(t) \to \frac{\bar{\alpha}_W}{2}$ and $\alpha_W(y(t)) \to \bar{\alpha}_W$.[7]

Therefore, profit per period approaches $\frac{\bar{\alpha}_W}{4}$ and deadweight loss per period goes to $\frac{\bar{\alpha}_W}{8}$.

Duopoly

When both Windows and Linux are available and Windows is sold at a price *p*, the customer precisely indifferent between the two, *q*, is given by:

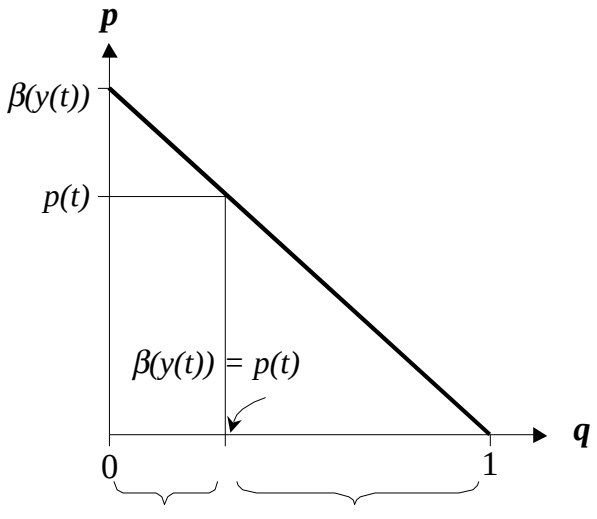
$$\alpha_W(y)(1-q) - p = \alpha_L(y)(1-q).$$

Thus, inverse demand is

$$p = \beta(y)(1-q)$$

where $\beta(y) \equiv \alpha_W(y) - \alpha_L(y)$ indicates the value difference between Windows and Linux.

Graphically,



[7] All proofs are contained in Appendix B.

Notice that in this model, customers in every new cohort are assumed to be myopic in the sense that they buy the OS that is immediately most valuable to them (after subtracting price). The extension to the case of forward-looking buyers is developed in Section IV.

We assume that at time $t = 0$, Windows is perceived as more valuable than Linux. That is, $\beta(y(0))>0$. Note that when Microsoft sets $p = 0$ and $\beta(y)>0$, demand for Windows is 1 (the size of the entering cohort). Thus, as long as $\beta(y)>0$, Microsoft can capture the entire new cohort by setting $p = 0$. However, in this case profit is also 0. The customer who most values Windows, is willing to pay no more than $\beta(y)$, an increasing function of market share y_W, for it. In contrast, if market shares are such that $\beta(y) \leq 0$ (Linux's perceived quality is at least as large as that of Windows), then nobody is willing to pay anything for Windows.

Because $y_w(t) = \int_0^t q(\tau)d\tau$ and $y_L(t) = \int_0^t (1 \pm q(\tau))d\tau$, we have

$$\dot{y} = q(t) - s(1 - q(t)) \tag{3}$$

Let r be the discount rate. Microsoft's problem is:

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t)\, p(t)\, dt \tag{4}$$

subject to

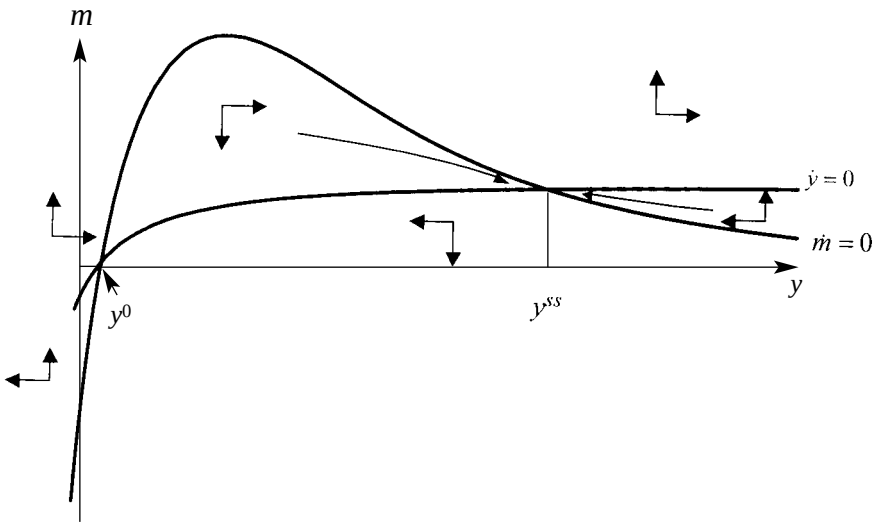
$$\dot{y} = q(t) - s(1 - q(t))$$
$$p(t) = \beta(y(t))(1 - q(t))$$
$$\beta(y(0)) > 0$$
$$p(t) \geq 0$$

We can therefore use standard phase diagram analysis to examine the long-run dynamics of competition.

Proposition 2. *Windows and Linux coexist in the long-run, steady-state equilibrium as long as* $s > 1$. *When* $s \leq 1$, *Windows pushes Linux out of the market.*

Proof. The proof involves using phase diagram analysis to graphically represent the path leading to steady state and show that path is optimal by checking Mangasarian's sufficient conditions. See Appendix B.

Proposition 2 implies that Microsoft is *never* pushed out of the market by the free, open source operating system, *regardless* of the speed of demand-side learning (the value of s), the difference in potential maximum values $(\bar{\alpha}_L - \bar{\alpha}_W)$ and market shares at time $t = 0$. The phase diagram looks as follows:



The two steady states y^0 and y^{ss} are characterized by

$$\beta(y^0) = 0$$

and

$$\frac{\beta'(y^{ss})}{\beta(y^{ss})} = \frac{r(s-1)}{s}, \tag{5}$$

respectively. The phase diagram reveals that y^0 is unstable and y^{ss} is a saddle point.

With the functional forms assumed in example 1, the two steady states can be computed explicitly. The unstable steady state is

$$y^0 = \frac{\alpha_L - \alpha_W}{\alpha_W}$$

and the saddle point is

$$y^{ss} = \frac{\alpha_L - 2\alpha_W}{2\alpha_W} + \sqrt{\frac{\alpha_L^2}{4\alpha_W^2} + \frac{s\alpha_L}{r(s-1)\alpha_W}}$$

Notice that $y^{ss} > y^0$. For example, when $\alpha_W = 2$, $\alpha_L = 3$, $r = 5\%$, and $s = \frac{9}{5}$, the steady states are $y_u^{ss} = \frac{1}{2}$ and $y^{ss} = 8$. In this example, if y ever fell below $\frac{1}{2}$ Windows would be perceived as less valuable than Linux. As long as $y(0) > \frac{1}{2}$, Microsoft's pricing strategy guarantees that $y(t)$ never falls below $\frac{1}{2}$ and thus Windows is not pushed out. The value of β in the steady state y^{ss} reflects the long-run difference in perceived quality between Windows and Linux.

The result that Windows will persist in the long run *regardless* of the difference between potential values of Windows and Linux (α_W and α_L) and regardless of the speed of demand-side learning on the part of Linux (*s*) is central to our inquiry on the competitive dynamics between open-source and closed software. Contrary to earlier results on competition with network externalities, in our model the failure of Linux to replace Windows is not due to switching or search costs (see, for example, David (1985)). Furthermore, the failure of a higher potential quality OS to eventually win the market out is not related to demand-side coordination issues (as in Farrell and Saloner (1985, 1986)) because demand coordination does not raise the instantaneous value of the OS on which buyers coordinate. In our model, without Microsoft's forward-looking pricing strategy, Windows would inevitably wind up being replaced by Linux (whenever $\alpha_L > \alpha_W$ and $s > 1$). Instead, it is Microsoft's strategic actions that generate the result. The market does not fully tip to Linux because Microsoft's strategic decisions prevent that from happening.

More generally, much of the network externalities literature focuses on one profit-maximizing firm versus another or, if looking at demand-side issues, assumes competitive supply. Instead, we look at asymmetric/mixed mode competition. In particular, the interaction between for-profit and not-for-profit entities seems particularly interesting in the context of knowledge development/innovation. In addition, our model features explicit dynamics, not a two-period abstraction as in much of the literature on network externalities.

It is interesting to notice that it is never optimal for Microsoft to "milk" its initial advantage $(\beta(y(0)) > 0)$ by setting high prices in the short term and at some future point leave the market. To understand why, notice first that the myopic profit-maximizing price in period *t* is $\rho = \frac{1}{2}\beta(y(t))$ (for an instantaneuous profit or $\pi(t) = \beta(y(t))$. Clearly, if Microsoft was determined to milk its short-term advantage, it would eventually set prices at this level (this would be Microsoft's optimal choice in the period immediately preceding its exit). Consider now a downward price deviation from this myopic (or "milking") profit-maximizing price to $p = \frac{1}{2}\beta(y(t)) - \delta$ where $\delta > 0$. The new profit in period *t* is $p = \frac{1}{2}\beta(y(t))$ which is strictly less than the myopic profit that period. However, by reducing price by δ, Windows is more valuable in period *t*+1 than if price had been maintained at the myopic level because quantity sold in period *t* is larger. In particular, $\beta(y(t+1)) = \beta\left(y(t) + \frac{1-s}{2} + \frac{\delta(1+s)}{\beta(y(t))}\right)$ instead of $\beta(y(t+1)) = \beta\left(y(t) + \frac{1-s}{2} + \frac{\delta(1+s)}{\beta(y(t))}\right)$. Now, if Microsoft sets $p = \frac{1}{2}\beta(y(t)) - \delta$ in period t and goes back to the myopic profit maximizir price in period *t*+1 $\left(p = \frac{1}{2}\beta(y(t+1))\right)$, the net present value of profit as a function of δ can be expressed as:

$$NPV(\pi(\delta)) = \frac{\beta(y(t))^2 - 4\delta^2}{4\beta(y(t))^2} + \frac{1}{4(1+r)}\beta\left(y(t) + \frac{1-s}{2} + \frac{\delta(1+s)}{\beta(y(t))}\right)$$

The derivative of $NPV(\pi(\delta))$ at $\delta = 0$ is:

$$\left.\frac{d(NPV(\pi(\delta)))}{d\delta}\right|_{\delta=0} = \underbrace{\frac{1}{4(1+r)}}_{+}\underbrace{\frac{d\beta(y(t+1))}{dy(t+1)}}_{+}\underbrace{\frac{1+s}{\beta(y(t))}}_{+} > 0$$

Therefore, regardless of the discount rate and regardless of the size of the initial advantage, it is always the case that, starting from the myopic profit-maximizing price, it is optimal to reduce current price a little bit. The reduction in instantaneous profit is more than offset by the corresponding increase in profit next period.

More precisely: the derivative of instantaneous profit evaluated at the myopic price $p = \frac{1}{2}\beta(y(t))$ is 0 (by definition). However, the period before Windows exits, it must be the case that $\beta(y(t)) \approx 0$ and because $q = 1 - \frac{p}{\beta(y(t))}$, we have that $-\frac{dq}{dp} \approx \infty$. Therefore, that last period lowering price a tiny little bit has a huge effect on next period's perceived value of Windows and almost no effect on the present period profit level. Thus, it is optimal for Windows to always set a lower price than the myopic optimum.

The following comparative statics are of interest:

Proposition 3 *Assume* s > 1, *then*

$$(a)\frac{dy^{ss}}{ds} < 0;\ (b)\ \frac{dy^{ss}}{dr} < 0;\ (c)\ y^0 < \lim_{s\to\infty} y^{ss} < \infty;\ (d)\lim_{s\to 1^+} y^{ss} = \infty;$$
$$(e)\ \lim_{r\to\infty} y^{ss} = y^0;\ (f)\ \lim_{r\to 0} y^{ss} = \infty.$$

That is,

(a) The larger Linux's demand-side learning, the smaller the steady-state difference in accumulated market shares between Windows and Linux. Once the steady state has been reached, y^{ss} is a constant but y_W and y_L keep growing without bound.

According to *(b)*, the more myopic Microsoft is, the more similar are the steady-state perceived qualities of Windows and Linux. And the more patient Microsoft is, the greater the long-term perceived quality advantage of Windows.

(c) Even as the sensitivity of y to y_L goes to infinity, there is a lower bound on y^{ss} strictly greater than y^0. As a consequence, if Windows is ahead, it will stay ahead *regardless* of the value of *s*.

(d) As Linux's demand-side learning approaches 1, if Windows is ahead, Linux will eventually be forced out of the market.

(e) The only case in which Microsoft is pushed out by Linux is if Microsoft is completely myopic. In this case, Microsoft goes after exactly 50% of every new cohort, but then if *s* > 1, Linux's effective cumulative market share will eventually become orders of magnitude larger than Windows'. In the long-run Windows is less valuable than Linux.

(f) If Microsoft became completely patient, it would effectively push Linux out of the market.

Having computed explicitly the stable steady state, it is trivial to calculate the implied quantities, prices, and profits, and to establish the following corollary:

Corollary *Microsoft's steady-state price and profit is always lower in a duopolistic industry structure.*

Clearly, at monopoly prices some customers prefer to get Linux for free. Microsoft takes this into account and lowers prices.

Cost

The most immediate way to overturn the result that Windows will prevail no matter the strength of Linux's demand-side learning is by introducing cost asymmetries. In particular, let *c* be a per-unit (marginal) cost for Windows and assume Linux's unit cost is zero. This can be thought of as production or sales cost, post-sales service, et cetera.

With *c*, we rewrite Microsoft's objective function as

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t)(p(t)-c)dt$$

and solve problem (4) with the exact constraints (see the proof of Proposition 2).

The presence of *c* jeopardizes Microsoft's ability to control the share of each entering cohort that is captured by Linux. As a consequence, if Linux's demand-side learning s is large enough, Linux can 'force' Windows out of the market.

It is also interesting to notice that with *c*, *s* does not need to be greater than 1 for Linux to be able to stay a viable competitor. If Microsoft's ability to build share is less than perfect, then y_L will increase relatively more rapidly and $s > 1$ is unnecessary for the viability of Linux. In the example, if $c = \frac{\alpha_W}{k}$, with $k > 1$, the maximum number of customers that Windows can get out of each cohort (if price is greater than or equal to marginal cost) is less than $\frac{k-1}{k}$. As $k \to 1$, Linux's period market share approaches 1.

Welfare

The corollary above suggests that a duopolistic industry structure is likely to dominate Microsoft's monopoly in terms of total welfare generation. Also, if Linux's potential quality is above that of Windows ($\alpha_L \geq \alpha_W$), one would expect that Linux's monopoly should dominate Windows' monopoly and the duopoly. In this section we analyze the welfare implications of each industry structure and show that neither of these claims is necessarily true.

We begin by analyzing welfare for the new cohorts entering after the steady state has been reached. The case where Microsoft is a monopolist is immediate. Total surplus (producer plus consumer surplus) is

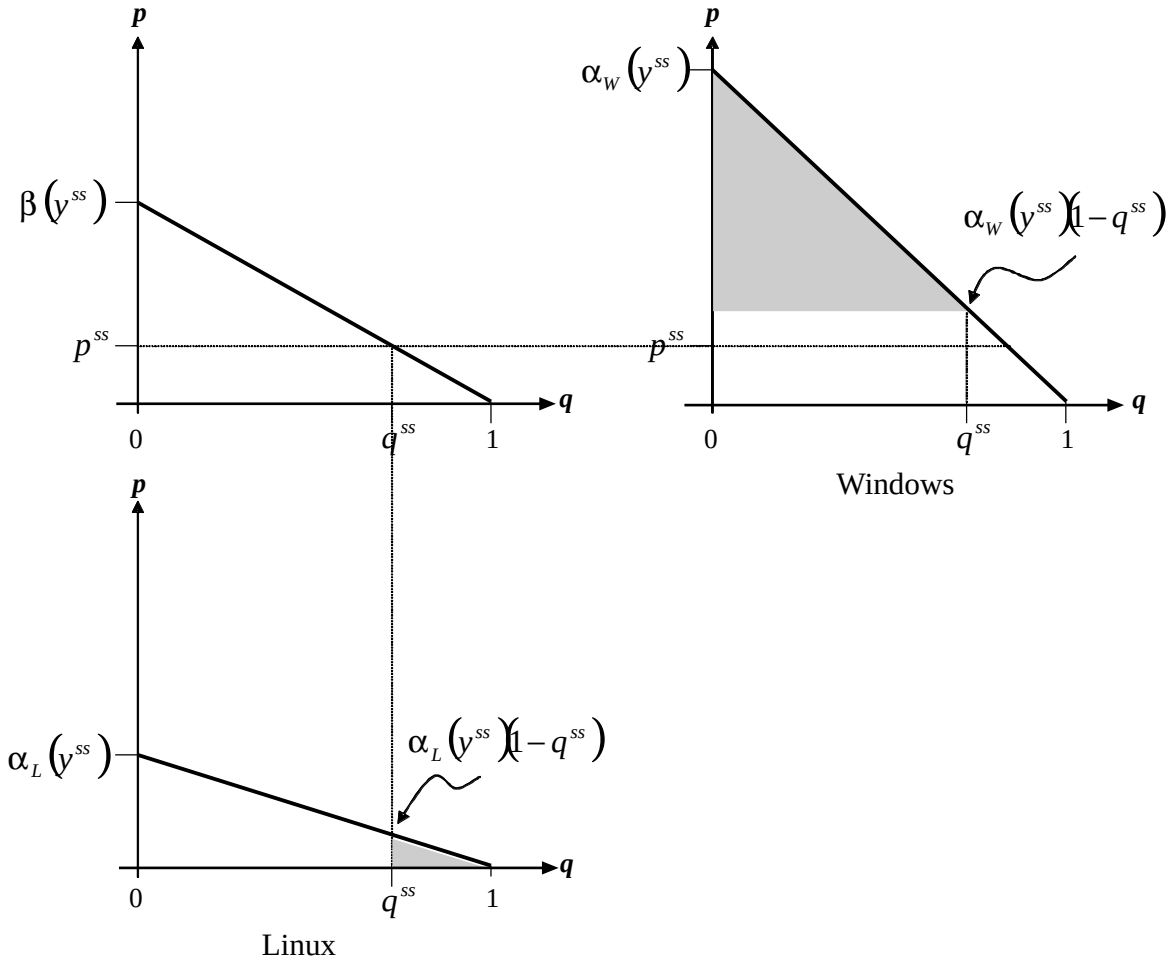
$$TS_W^{Monopoly} = \frac{3}{8}\alpha_W$$

Similarly, Linux's monopoly steady-state total surplus is

$$TS_L^{Monopoly} = \frac{1}{2}\alpha_L$$

Thus, Linux's monopoly is socially more desirable than a Windows' monopoly as long as $\alpha_L > \frac{3}{4}\alpha_W$.

More interesting is the comparison between a Windows monopoly and a Windows-Linux duopoly. The following figure summarizes the computation of total surplus generated by the duopoly (total surplus is the sum of the shaded areas):



Using equations (A1) and (A9) in Appendix B, it can be easily shown that total surplus generated by Windows is given by

$$TS_W^{Duopoly} = \frac{\alpha_W(y^{ss})}{2}\frac{(s+1)^2-1}{(s+1)^2},$$

and that of Linux by

$$TS_L^{Duopoly} = \frac{\alpha_L(y^{ss})}{2(s+1)^2}.$$

Total duopoly surplus is

$$TS^{Duopoly} = \frac{1}{2(s+1)^2}\left(\alpha_W(y^{ss})\left((s+1)^2-1\right)+\alpha_L(y^{ss})\right).$$

Whether $TS^{Duopoly}$ is greater or less than $TS_W^{Monopoly}$ is ambiguous. There are two reasons why duopoly can enhance consumer surplus. First, because Linux is available, Microsoft is induced to set lower prices. Second, those individuals in the cohort who do not buy Windows are not left empty-handed, they can download and use Linux for free and this raises total surplus. However, the fact that part of the population uses Linux lowers the perceived value of Windows (because there is some substitution of third-party complement development from Windows to Linux). If this effect is large, monopoly (where all developers produce complements for Windows) may result in larger total surplus.

To see formally that the comparison between $TS^{Duopoly}$ and $TS_W^{Monopoly}$ is ambiguous, suppose first that for some technological trajectories ($\alpha_W(.)$ and $\alpha_L(.)$) we have that $TS^{Duopoly} > TS_W^{Monopoly}$. Consider now a new technological trajectory $\alpha^*_W(y)$ that coincides with $\alpha_W(y)$ everywhere up to a point $y^* > y^{ss}$. From that point on, $a^*_W(y)$ grows linearly (with slope $\left.\frac{d\alpha_W(y)}{dy}\right|_{y=y^*}$) up to a magnitude larger than $\bar{\alpha}_W$. From that point on, $\alpha^*_W(y)$ levels off. Clearly, with this new technological trajectory, $TS_W^{Monopoly} > TS^{Duopoly}$.

Let's now consider the case in which for some technological trajectories $TS_W^{Monopoly} > TS^{Duopoly}$. Let $r \to 0$. $TS_W^{Monopoly}$ does not change. However, by Proposition 3(f), $y^{ss} \to \infty$ and thus $\alpha_W(y^{ss}) \to \bar{\alpha}_W$. Furthermore, by (A1) and (A9), $p^{ss} = \beta(y^{ss})\frac{1}{s+1}$. These two facts together imply that $p^{ss} \to \bar{\alpha}_W \frac{1}{s+1}$. Therefore, because in the limit both demand functio are the same and with a duopoly Windows is sold at a lower price, it is the case that $TS_W^{Monopoly} < TS^{Duopoly}$.

Therefore,

Proposition 4 *Steady-state total surplus may be larger under Windows' monopoly than under duopoly.*

We conclude this section with an observation about welfare on the path to the steady state. As mentioned above, when $\bar{\alpha}_L \geq \bar{\alpha}_W$, steady-state total welfare under Linux monopoly is larger than steady-state total welfare under Windows monopoly or duopoly. However, if it will take a long time for Linux to build market share, Linux's monopoly may still not maximize the net present value of total surplus. More precisely, let ρ be the regulator's discount rate and $TS_W^{Monopoly}(t)$, $TS^{Duopoly}(t)$, and $TS_L^{Monopoly}(t)$, be period t's total surpluses under Windows monopoly, duopoly, and Linux monopoly, respectively. Suppose that users die at rate δ; that is, if M users are alive in period t, then δM users will be alive in period t+1 (in addition to the new entrants). The net present value of total surplus can then be expressed as:

$$TS_k^i = \int_0^{\infty}\left(\int_0^t e^{-\frac{1-\delta}{\delta}(t-\upsilon)} TS_k^i(\upsilon)d\upsilon\right)e^{-\rho t}dt,$$

for $i \in \{Monopoly, Duopoly\}$ and $k \in \{W, L, \emptyset\}$.

Although it is not possible to get explicit price sequences on the path to the steady state, it is immediate to conclude that Linux's monopoly may not maximize net present value of welfare, the reason being that if $\alpha_L(y(0))$ is substantially lower than $\alpha_W(y(0))$ and ρ is high (so that future welfare enhancements are discounted heavily), then the immediate larger surplus generated by a Windows monopoly outweighs the potential future gains associated with Linux.

IV. Extensions

We now present three simple extensions to the basic model: forward-looking buyers, strategic commitment to Linux and piracy of Windows. Forward-looking buyers may play to the advantage of Microsoft, and strategic commitment helps Linux build market share, but may or may not be welfare-enhancing. Even less expectedly, piracy helps Windows increase its steady-state quality difference, and may, therefore, result in increased profits for Microsoft.

Forward-looking buyers

Let ϕ be the (common) discount rate used by buyers to evaluate future utility. Suppose that present time is t, that the state variable has value $y(t)$ and that price is p. The threshold buyer q is found by solving:

$$\max_q \left[\tfrac{1}{2} \int_{\tau=t}^{\infty} e^{-\phi(\tau-t)} \left(\alpha_W(y(\tau)) - (1-q)^2 \beta(y(\tau)) \right) d\tau - pq \right] \tag{9}$$

(To see this, compute consumer surplus for given p and notice that individual utility maximization is equivalent to total surplus maximization.) Notice that when $\phi = \infty$, (9) reduces to $q = 1 - \frac{p}{\beta(y(t))}$, just as in the benchmark model with myopic buyers.

We analyze the case in which buyers are forward-looking but believe that they are so insignificant that their purchase decision will not affect the state variable. With this assumption, (9) can be rewritten as:

$$\max_q \left[-\tfrac{1}{2} \int_{\tau=t}^{\infty} e^{-\phi(\tau-t)} (1-q)^2 \beta(y(\tau)) d\tau - pq \right].$$

Solving the program yields:

$$q = 1 - \frac{p}{\int_{\tau=t}^{\infty} e^{-\phi(\tau-t)} \beta(y(\tau)) d\tau} \quad . \tag{10}$$

Comparing (10) and the demand function in the benchmark model ($q = 1 - \frac{p}{\beta(y(t))}$), we see that whether the threshold q with forward-looking buyers is larger or smaller than that with myopic buyers depends on the buyers' view on which OS will be more valuable in the future. In particular, when $\beta(y(t))$ is large and positive and buyers' discount rate is not too low, the presence of forward-looking buyers plays to the advantage of Microsoft because for a given p, the threshold q is now larger than with myopic buyers. However, if buyers expect $\beta(y(t))$ to eventually turn negative and their discount rates are low, Microsoft will be forced to price lower than in the case where buyers are myopic.

Equation (10) shows that it may be worth for Microsoft to influence the value of $\int_{\tau=t}^{\infty} e^{-\phi(\tau-t)} \beta(y(\tau)) d\tau$ by infusing "fear, uncertainty, and doubt" into the OS user community. Such emotions were stirred in the Linux community by, among other things, SCO, a small Swiss-based "vulture" firm that had bought up the intellectual property rights to a particular version of Unix and was threatening Linux users with lawsuits over infringement of those rights unless they agreed to pay it substantial licensing fees. IBM, which was one of the

prime corporate sponsors of Linux as well as the target of a lawsuit by SCO that sought $1 billion in damages, alleged in mid-2003 that SCO was in cahoots with Microsoft[9].

However, regardless of the value of $\int_{\tau=t}^{\infty} e^{-\phi(\tau-t)}\beta(y(\tau))d\tau$, Microsoft can once again guarantee a 100% market share of every new cohort by pricing sufficiently low. Therefore, the result that Microsoft is not pushed out of the market by Linux (regardless of the intrinsic advantages of Linux) remains intact when buyers are forward-looking.

Strategic Commitment to Linux

Strategic commitment to Linux may manifest itself in governmental procurement decisions, in decisions by strategic buyers such as IBM with a direct economic interest in opposing Microsoft, and so on. For example, our qualitative attempt to classify the governments in several dozen countries on the basis of the software platform that they supported identified 13 countries, with the European Community particularly well-represented, that seemed to be in the Linux camp, versus five in the Microsoft camp and 12 in which governments seemed to have made relatively clear commitments to both platforms (see Appendix C for details). Note that some degree of intrinsic affinity might be expected between the open-source movement and governments in the presence of shared nonprofit objectives; in addition, government procurement seems, in this category as in many others, more price-sensitive, on average, than private or at least corporate procurement.

To model such effects, suppose that a measure $\varepsilon > 0$ of customers in every cohort are committed to Linux for some reason of this sort. It is important to identify those customers. We distinguish two polar cases. First, the potential customers represented by ε would have used Linux even if they were not committed to Linux. In this case, there is no change. The steady state y^{ss} is as given by equation (5).

Second, suppose that these ε customers would all have bought Windows had they not been committed to Linux. These are individuals that value Windows above Linux (after subtracting p) but they use Linux instead. We now examine the effect of such commitment on long-run competitive dynamics.

Recall that, absent strategic commitment to Linux, demand for Windows is

$$q = 1 - \frac{p}{\beta(y)}$$

But because ε 'would-be-buyers' of Windows are committed to Linux instead, demand for Windows is

$$q = 1 - \frac{p}{\beta(y)} - \varepsilon \qquad \text{or} \qquad p = (1 - q - \varepsilon)\beta(y)$$

The equation of motion of y is as before (eq. 3), $y = q(t) - s(1 - q(t))$.

---

9 Andrew Dolley [2003].

Microsoft's problem is

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t)\, p(t)\, dt$$

subject to

$$\dot{y} = q(t) - s(1 - q(t))$$

$$q(t) = 1 - \frac{p(t)}{\beta(y(t))} - \varepsilon$$

$$\beta(y(0)) > 0$$

$$p(t) \geq 0$$

Solving for the unique saddle point steady state y^{ss} (see proof of Proposition 2), we obtain

$$\frac{\beta'(y^{ss})}{\beta(y^{ss})} = \frac{s - 1 + \varepsilon(1+s)}{1 - \varepsilon(1+s)} \frac{r}{s} \tag{12}$$

Strategic commitment to Linux by a portion ε of Windows' 'would-be-buyers' has two effects: it helps build market share of Linux, thus increasing its value vis a vis Windows, and it also forces Microsoft to lower its prices for Windows. A simple computation using eq. (12) yields:

$$\frac{dy^{ss}(\varepsilon)}{d\varepsilon} = \frac{\beta(y) r s (1+s)}{\underbrace{(-1 + \varepsilon(1+s))}_{-} \left(\underbrace{\beta'' s(-1 + \varepsilon(1+s))}_{+} + \underbrace{\beta' r(-1 + \varepsilon(1+s) + s)}_{+} \right)} < 0,$$

a negative number for ε small.

The following proposition shows that with the presence of strategic buyers, if demand-side learning on the part of Linux is sufficiently swift, Windows is pushed out.

Proposition 6 *For given ε, if s is sufficiently large, Microsoft is pushed out of the market. Equivalently, for given s, if ε is sufficiently large, Microsoft is pushed out of the market.*

Intuitively, when *s* is large, Microsoft has to make sure that Linux's share (*1-q*(*t*)) remains very small. However, the presence of a portion of potential customers who will never buy Windows jeopardizes Microsoft's ability to capture (current cohort) market share. If such ability is sufficiently damaged by these strategic buyers, Windows is eventually pushed out.

Therefore, if *s* is large, strategic commitment to Linux induces efficient push-out of Windows by Linux. However, if *s* is small so that, without strategic commitment, Linux would be efficiently pushed out by Microsoft, with strategic commitment Linux may prevail.

We conclude that strategic commitment to Linux may enhance total welfare if *s* is large, because that may push Windows out when it is efficient to do so. But strategic commitment to Linux may reduce total welfare if *s* is very small because then Linux will not be pushed out even though its push-out would increase welfare.

Piracy of Windows

We can also use an extension of the model developed earlier to analyze the effects of piracy, as in the illegal copying, distribution, and use of software. Of course, because Linux is free, it is only meaningful to talk about piracy of Windows.

Suppose that every period, a portion ρ of the entering cohort pirates Windows. We assume that the portion of pirates is small (positive but sufficiently close to zero for an interior solution to exist). Again, it is important to identify who these customers are. Suppose a portion $\mu \in (0,1)$ comes from individuals who have bought Windows (high-value customers) and the rest, $1-\mu$, would have gotten Linux (low-value customers).

Because $\rho\mu$ 'would-be buyers' pirate Windows, demand for Windows at price p is

$$q = 1 - \frac{p}{\beta} - \rho\mu \ .$$

Notice that $y_W = q + \rho$ and $y_L = 1 - q - \rho$. Therefore,

$$\dot{y} = q + \rho - s(1 - q - \rho)$$

Microsoft's problem is:

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t) p(t) dt$$

subject to

$$\dot{y} = q(t) + \rho - s(1 - q(t) - \rho)$$

$$q(t) = 1 - \frac{p(t)}{\beta(y)} - \rho\mu$$

$$\beta(y(0)) \geq 0$$

$$p(t) \geq 0$$

Solving the program, the unique saddle point steady state is characterized by:

$$\frac{\beta'(y^{ss})}{\beta(y^{ss})} = \frac{r(\rho(s+1)(\mu-2) + s - 1)}{(\rho(1+s) - s)(\rho(1+s)(\mu-1) - 1)}. \qquad (13)$$

Let $A \equiv (\rho(1+s) - s)(\rho(1+s)(\mu-1) - 1)$ and $B \equiv \rho(s+1)(\mu-2) + s - 1$.

Differentiating (13) implicity, we see that

$$\frac{dy^{ss}(\rho)}{d\rho} = -\frac{\beta(y) r(1+s)((\mu-1)(\rho(1+s)(B+s-1) - s^2) + 1)}{A(\beta''(y)A - \beta'(y)rB)}$$

which is positive (when ρ is small).

An increase in piracy ρ has two effects. On the one hand, if $\mu < 1$, some people that would have chosen Linux, now use Windows because they get the OS for free. So, there is an increase in period market share of size $\rho(1-\mu)$. On the other hand, if $\mu > 0$, some people who would have bought Windows now get Windows for free. This does not affect instantaneous market share, but shrinks demand and Microsoft is induced to set lower prices and, as a consequence, the steady-state market share differential also increases.

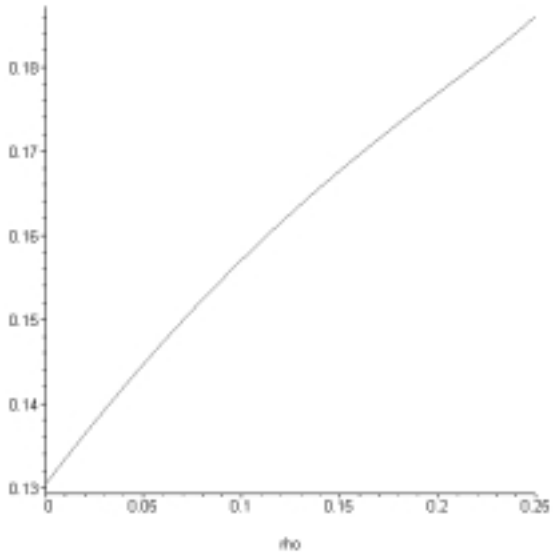
Finally,

$$\frac{dy^{ss}(\mu)}{d\mu} = \frac{\beta(y)r\rho(1+s)(\rho(1+s)-s)}{(\rho(1+s)(\mu-1)-1)(\beta''(y)A-\beta'(y)rB)}$$

is negative (for ρ small). The larger the piracy by 'would-be buyers', the less the increase in instantaneous market share (as compared to a situation where piracy comes from individuals who would have used Linux).

The following example shows that piracy may be beneficial to Microsoft not only in terms of larger steady-state market share, but also in terms of steady-state profit.

Example 2. Assume the functional forms in Example 1 and set s = 3, r = 10%, μ =.3, $\alpha_W = 1$, and $\alpha_L = 2$. Solving for the steady-state profit as a function of piracy ρ, we have:



Some Evidence

We compiled data on total Windows and Linux shipments for a sample of 51 countries in 4 continents and ran the following correlations:

	(B) Nominal GDP (US$ at PPP) per Capita	*(D) Population (millions)*	*(G) Linux share*	*(H) Linux/MS ratio*	*(I) Total Shipments*	*(I/B) Total Shipments/GDP*	*(I/D)Total Shipments/Pop*	*(L) Piracy Rates (%) 2000*	*(N) Growth Competitive Index 2001*	*(O) Technology Index 2001*	*(P) Innovation subindex 2001*	*(Q) ICT subindex 2001*
(B) Nominal GDP (US$ at PPP) per Capita	1.00											
(D) Population (millions)	-0.33	1.00										
(G) Linux share	0.35	-0.23	1.00									
(H) Linux/MS ratio	0.37	-0.27	0.89	1.00								
(I) Total Shipments	0.11	0.09	-0.23	-0.10	1.00							
(I/B) Total Shipments/GDP	-0.03	0.45	-0.30	-0.20	0.91	1.00						
(I/D)Total Shipments/Pop	0.18	0.00	-0.22	-0.12	0.96	0.84	1.00					
(L) Piracy Rates (%) 2000	-0.67	0.37	-0.43	-0.50	-0.08	0.11	-0.11	1.00				
(N) Growth Competitive Index 2001	0.65	-0.21	0.27	0.25	0.08	0.00	0.15	-0.75	1.00			
(O) Technology Index 2001	0.57	-0.32	0.32	0.29	0.04	-0.08	0.10	-0.77	0.92	1.00		
(P) Innovation subindex 2001	0.70	-0.31	0.43	0.39	0.07	-0.06	0.14	-0.74	0.87	0.89	1.00	
(Q) ICT subindex 2001	0.65	-0.37	0.42	0.42	0.06	-0.07	0.14	-0.84	0.92	0.91	0.86	1

Notice that just as the results above show, piracy rates (row L) are negatively associated with Linux penetration (column G) and the Linux/Microsoft ratio.

V. Conclusions

The specification, analysis and (very partial) testing of the model of dynamic mixed duopoly in Sections III and IV of this paper was informed in multiple ways by the case of Linux vs. Windows that was described in Section II. The case suggested a specific way of framing the interactions between open-source software development initiatives and their for-profit competitors: in terms of competition between an open-source product priced at zero and a for-profit "closed" product, in the presence of demand-side learning effects. Accordingly, this paper set up and analyzed a highly stylized model of this sort in order to characterize the equilibrium outcomes to dynamic competition of this sort and some of their comparative static and welfare properties. The analysis, in particular, while not game-theoretic in the usual sense of interdependent strategy choices, was strategic in the sense of requiring Microsoft Windows to take a deep look into the future that recognized intertemporal linkages in its profit function (e.g., between past or current choices and future profits) as opposed to acting myopically, which would have led to very different predicted outcomes. As Arrow [1964] and others have stressed, such *intertemporal* linkages and the commitment or irreversibility underlying them are the key reason that the optimal intertemporal investment program may not coincide with the *instantaneous* equation of the marginal productivity and the marginal cost of capital (of whatever sort), i.e., the myopic investment program.

Embedding irreversibility in the form of sticky resources in a formal analytical model of mixed duopoly yielded some arguably surprising conclusions. Thus, Microsoft Windows' persistence exceeded our pre-analytic intuitions because of the effects of Microsoft's strategic management of its position relative to Linux Other effects/possibilities, e.g., that strategic procurement of Linux could hurt welfare, were somewhat unexpected. And other effects were surprising for other reasons, such as the apparent first-order effect of piracy rates on Linux penetration.

More broadly, the present paper can also be read as an attempt, under admittedly specializing assumptions, to analyze interactions between for-profit and not-for-profit competitors. We know very little about competitive interactions in such a setting even though, as Section I indicated, mixed objective functions of this sort can be thought of as covering a very broad range of situations. This paper has made a very modest start at addressing this large gap in our knowledge.

Appendix A: Prior Research on Open-Source Software

Much of the prior literature on Linux/open-source focuses on how open-source development efforts are organized, particularly (as noted above) the satisfaction of the individual rationality or participation constraints of user-developers who are critical to learning on the demand side. For compactness, we will refer to this as the organizational strand of research on open-source.

The most focused substrand of organizational research on open-source has assumed utility-maximizing (potential) users/developers and has tried to derive various sorts of comparative static predictions about their developmental contributions. In early work of this sort, Thorn and Connolly [1987] used theories of the economics of public goods to argue that the rates and effectiveness of discretionary information-sharing amongst employees in an organization would tend to decrease as (1) participation costs increased; (2) the size of the overall group increased; (3) the value of information to participants decreased; and (4) the asymmetries in information values and benefits across participants increased.

More recent work in this line has pushed farther with formalizing these insights and developing new ones. Thus, Kuan [1999] framed consumer choice between open and closed-source software as a make-or-buy decision, with the former option entailing a further decision about how much effort to exert contributing to the quality of software (a public good), and concluded that the advantages of open-source software were higher for programmers than nonprogrammers. He also inferred that if most high-paying users were also programmers, open-source or community organization would be more likely (e.g., engineering tools or utilities), whereas if most high-paying users were nonprogrammers, proprietary or closed organization would be more probable (e.g., word processors, spreadsheets and other products with a broad non-engineering market). Bessen [2002] also analyzed a self-selection model with consumers helping test and debug different variants of a complex product with many (interacting) features, of which only a fraction might be valuable to any particular user. He concluded that given open source, individual users who placed a high enough value on the product would test and debug their own use-product and that, as long as costs were sufficiently low and product complexity sufficiently high, more use-product combinations would be tested with open source than under the proprietary case, a larger market would be served, and social welfare would be higher. Johnson [2002] analyzed a self-selection model with various informational imperfections and concluded that whether open-source development would increase when applications had a modular structure depended on whether the developer base exceeded a critical size; he also provided some finite and asymptotic results of effects of changing the population size of user programmers and tried to explain why certain useful programs don't get written. Xu [2002], with a variant of the same basic model, showed that decreasing open-source development costs need not necessarily increase the amount of open-source software development, and proposed several other counterintuitive results as well. And so on.

A second substrand of the organizational literature has looked somewhat more broadly at whether open-source software development efforts can be explained as the outcome of private cost-benefit analysis by user-developers or whether other, less conventionally economic motivations—e.g., altruism, participation in a gift economy/culture in which social status depends more on what one gives away than what one possesses, or even a visceral dislike of Microsoft—need to be invoked to explain private provision of the public good of improved software quality. Thus, Lerner and Tirole [2002] argued that

Appendix A (continued)

conventional cost-benefit analysis may be sufficient once one accounts for benefits related to career concerns and ego gratification (stemming from peer recognition) that induce an incentive for an individual to signal high quality through participation in open-source development. They also suggested that signaling incentives might be strengthened in open-source environments by better performance measurement (given the care with which individual contributors tend to be credited), full initiative by (empowered) programmers, and greater labor market fluidity/knowledge portability, and that other factors favorable to open source include modularity, the existence of "fun" challenges and credible leadership. And, in a similar vein, Lakhani and von Hippel [2002] looked—in the context of Apache—at the performance of the mundane but essential task of providing high-quality field support (to overcome either defects in the product or deficiencies in the user's understanding) and concluded that the need for explanations such as altruism and even (delayed) signaling benefits was limited by the inference that most of the effort information-providers expended could be understood in terms of the direct rewards they derived immediately, i.e., in terms of learning for themselves.

A third, more miscellaneous substrand of the organizational literature on open-source has taken the even broader, more inductively-oriented approach of describing the actual organization of such software development efforts. Thus, three of the core contributors to the development of Apache, Mockus, Fielding and Herbsleb [2000], built on their experience of that project, as well as the history of others, including Linux, to offer some rough numerical requirements for their organization: a core group of developers, no larger than 15 people, to control the approval and integration of new/modified code into the ongoing stream of "official" releases—a process more centralized than most others in open-source development—and to create more than 80% of new functionality, strict code ownership policies to disaggregate open-source efforts that would otherwise be too large, a group larger by an order of magnitude than the core group to repair defects, and a group another order of magnitude larger yet to report problems. And the governance of such projects and, specifically, the legal tactics employed to protect the public property that they create, are discussed by O'Mahony [2003].

But these and other organizational issues surrounding open source, while undeniably interesting, are far from the only ones of interest. A second distinct set of issues concerns the outcomes to and implications of competition between a not-for-profit open standard priced at zero (e.g., Linux) and a for-profit closed standard (e.g., Windows). This competitive strand of research on open-source software is much less developed than the organizational strand discussed above, even though the rhetoric about it—open source as innovation savior vs. destroyer—can get quite heated. While papers that focus on the relative efficiency of open and closed development models are obvious reference points, they generally neglect interactions between the two models and the effects of moving late vs. early in determining competitive outcomes (e.g., Kogut and Metiu [2001]). Still, some specific analytical contributions are worth noting. Bitzer [2000] proposed a simple model to make the point that the less the heterogeneity in product space between open-source and closed software, the more likely competition is to collapse prices below the levels necessary to support the proprietary software developer's (higher) average costs and lead it to abandon its development efforts. Dalle and Jullien [2002] employed a simulation approach to establish that increasing returns associated with creativity and their (re)distribution toward end-users could create global and local positive externalities strong enough to help Linux reverse

Appendix A (continued)

current standardization on Windows 2000. And Schmidt and Schnitzler [2002] set up a simple, essentially static model of Hotelling-like horizontally differentiated competition between an open-source product and a for-profit closed product and showed that, within that setup, forced (by the government) procurement of the open-source product would unambiguously reduce welfare.

None of these papers, however, really embeds the competition between Linux and Windows in a dynamic model with demand-side learning of the sort suggested by the previous section. Such a model is presented in Section III of this paper.

Appendix B: Proofs

Proof of Proposition 1. Because Windows is a monopolist, demand is linear, and unit cost is zero, every period $q \geq \frac{1}{2}$. But then $y_W(t) \to \infty$ and $\alpha_W(y_W(t)) \to \overline{\alpha}_W$. Therefore, in the steady state, demand is $p = \overline{\alpha}_W(1-q)$.

Proof of Proposition 2. The following three properties of $\beta(y) \equiv \alpha_W(y) - \alpha_L(y)$ will be used in what follows. First, β(y) is increasing for all *y*:

$$\frac{d\beta(y)}{dy} = \underbrace{\frac{d\alpha_W(y)}{dy}}_{+} - s\underbrace{\frac{d\alpha_L(y)}{dy}}_{-} > 0 \cdot$$

Second,

$$\lim_{y\to\infty} \beta(y) = \lim_{y\to\infty} \alpha_W(y) - s\lim_{y\to\infty} \alpha_L(y) = \overline{\alpha}_W \cdot$$

Finally, $\beta(y)$ is concave for $y > y^0$ (assumption 4 in Section III).

Microsoft's problem is

$$\max_{p(t)} \int_0^\infty e^{-rt} q(t) p(t) dt$$

subject to

$$\dot{y} = q(t) - s(1 - q(t))$$

$$q(t) = 1 - \frac{p(t)}{\beta(y(t))}$$

$$y(0) \geq 0$$

$$p(t) \geq 0$$

Because we assume that Windows has the advantage in terms of perceived value at time $t = 0$, the case in which $s \leq 1$ is trivial as Linux can never even get started. To analyze the more interesting case where Linux's demand-side learning is superior, we use standard dynamic programming. Assume $s > 1$, the current-time Hamiltonian is[10]:

$$H = \left(1 - \frac{p}{\beta(y)}\right)p + m\left(\left(1 - \frac{p}{\beta(y)}\right)(1+s) - s\right) \cdot$$

[10] For notational simplicity, we omit time dependence of *y*, *m* (the Hamiltonian multiplier), and *p*.

The first-order conditions are

$$\frac{\partial H}{\partial p} = 0 \Rightarrow p = \frac{1}{2}(\beta(y) - m(1+s)), \tag{A1}$$

$$mr - \frac{\partial H}{\partial y} = \dot{m} \Rightarrow \dot{m} = mr - \frac{\beta'(y)p(p + m(1+s))}{\beta(y)^2}, \tag{A2}$$

and

$$\frac{\partial H}{\partial m} = \dot{y} \Rightarrow \dot{y} = \left(1 - \frac{p}{\beta(y)}\right)(1+s) - s \quad . \tag{A3}$$

Using (A1), we may rewrite (A2) and (A3) as

$$\dot{m} = mr - \frac{\beta'(y)\left(\beta(y)^2 - m^2(1+s)^2\right)}{4\beta(y)^2} \tag{A4}$$

and

$$\dot{y} = \frac{\beta(y)(1-s) + m(1+s)^2}{2\beta(y)} \tag{A5}$$

Now, (A4) and (A5) form a system of differential equations in m and y. Consider the following two equations

$$0 = mr - \frac{\beta'(y)\left(\beta(y)^2 - m^2(1+s)^2\right)}{4\beta(y)^2} \tag{A6}$$

and

$$0 = \beta(y)(1-s) + m(1+s)^2. \tag{A7}$$

The steady states are all those pairs $\langle m, y \rangle$ that satisfy simultaneously equations (A6) and (A7). We now isolate m in (A6) and (A7) and investigate the shape of the resulting functions. Solving (A6) for m, we have

$$m_{\dot{m}=0} = \frac{\beta(y)\left(-2r\beta(y) \pm \sqrt{4r^2\beta(y)^2 + (\beta'(y))^2(1+s)^2}\right)}{\beta'(y)(1+s)^2}$$

There are two cases, one for each sign of the square root. Consider first

$$m_{\dot{m}=0} = \frac{\beta(y)\left(-2r\beta(y) + \sqrt{4r^2\beta(y)^2 + (\beta'(y))^2(1+s)^2}\right)}{\beta'(y)(1+s)^2} \tag{A8}$$

Notice that (A8) is equal to 0 at y^0 such that $\beta(y^0) = 0$ and it is strictly positive for all $y > y^0$.

Solving (A7) for m, we have

$$m_{\dot{y}=0} = \frac{\beta(y)(s-1)}{(1+s)^2} \tag{A9}$$

Equation (A9) equals 0 at y^0 and it is strictly positive for all $y > y^0$. Taking the second derivative, we see that $m_{\dot{y}=0}$ is a concave function of y.

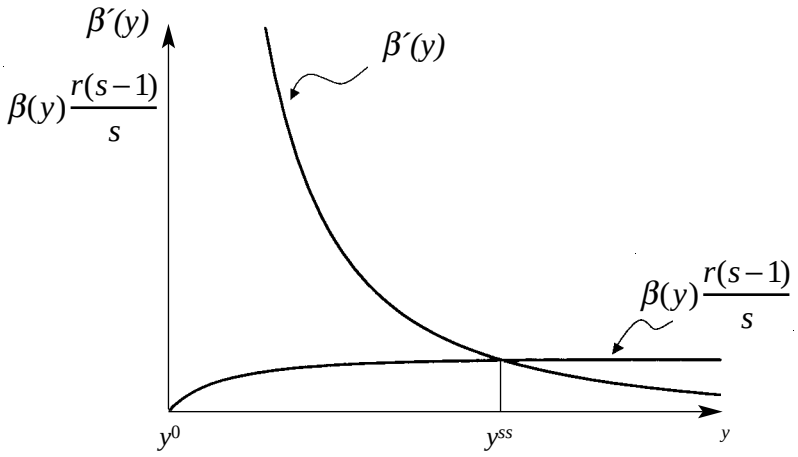
Equating (A8) and (A9) we see that there are two steady states. First, because

$$m_{\dot{m}=0}(y^0) = m_{\dot{y}=0}(y^0)$$

y^0 is a steady state. In this steady state, Windows is out. If $y = y^0$ is ever reached, the only way for Windows to stay in is by setting $p(t) = 0$ forever after. Second, it is easy to simplify $m_{\dot{m}=0}(y) = m_{\dot{y}=0}(y)$ to get

$$\frac{\beta'(y)}{\beta(y)} = \frac{r(s-1)}{s} \tag{A10}$$

Let y^{ss} satisfy equation (A10). Because the function $\beta(y)\frac{r(s-1)}{s}$ is always increasing and $\beta'(y)$ always decreasing, y^{ss} is unique:



Notice that

$$\left.\frac{dm_{\dot{m}=0}}{dy}\right|_{y=y^0} = \frac{\beta'(y^0)}{1+s} > \frac{\beta'(y^0)(s-1)}{(1+s)^2} = \left.\frac{dm_{\dot{y}=0}}{dy}\right|_{y=y^0}$$

Therefore, $m_{\dot{m}=0}$ has larger slope at y^0 than $m_{\dot{y}=0}$. In addition,

$$\lim_{y\to\infty} m_{\dot{m}=0} = 0 < \frac{\overline{\alpha}_W (s-1)}{(1+s)^2} = \lim_{y\to\infty} m_{\dot{y}=0}$$

Thus, at y^{ss}, $m_{\dot{m}=0}$ cuts $m_{\dot{y}=0}$ *from above*. (We use this fact to determine the stability of y^{ss}.) Because $y^{ss} > y^0$, in this steady state Windows survives in the sense that Microsoft needs not charge zero price in order to survive.

To sketch the directions of movement compatible with equations (A4) and (A5), we first consider the locus $\langle m, y \rangle$ where $\dot{m} = 0$. At a point $\langle m_A, y_A \rangle$ on the locus (A6), $\dot{m} = 0$ is satisfied. At a point $\langle m_A + k, y_A \rangle$, $k > 0$, *above* the locus, we have

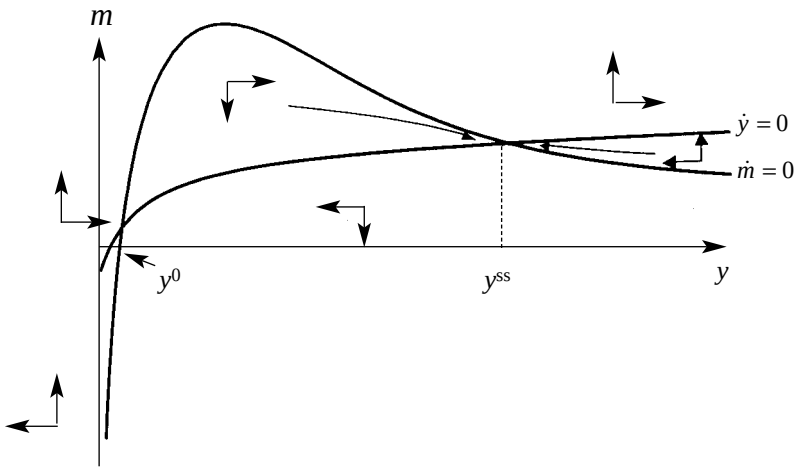
$$(m_A + k)r - \frac{\beta'(y_A)\left(\beta(y_A)^2 - (m_A + k)^2(1+s)^2\right)}{4\beta(y_A)^2} > 0$$

thus, $\dot{m} > 0$. Similarly, we see that m is decreasing at points below the $\dot{m} = 0$ locus. Next, consider the points for which $\dot{y} = 0$ At a point $\langle m_B, y_B \rangle$ on the locus (A7), $\dot{y} = 0$ is satisfied. At a point $\langle m_B + k, y_B \rangle$, $k > 0$, *above* the locus, we have

$$\beta(y_B)(1-s) + (m_B + k)(1+s)^2 > 0$$

thus, $\dot{y} = 0$. Similarly, it is easy to see that y is decreasing at points below the $\dot{y} = 0$ line.

The following phase diagram summarizes the analysis:



Steady state y^{ss} is a saddle point and y^0 is unstable. (See Kamien and Schwartz, 1991, page 178, cases (i) and (d), respectively). To see that the path leading to steady state y^{ss} is optimal, one can easily check Mangasarian's sufficient conditions. (See Seierstad and Sydsæter, Theorem 12, page 234.)

We finally have to consider the case in which the solution to solving (A6) for *m* is

$$m_{\dot{m}=0} = \frac{\beta(y)\left(-2r\beta(y) - \sqrt{4r^2\beta(y)^2 + (\beta'(y))^2(1+s)^2}\right)}{\beta'(y)(1+s)^2}$$

Replicating the above analysis yields a unique steady state at y^0. This is now a saddle point. However, phase diagram analysis reveals that along the path to the steady state, the Hamiltonian multiplier $m(t)$ (which is the marginal valuation of the state variable $y(t)$ at each moment in time) is *always* negative. Thus, along this path, an increase in the difference between the perceived quality of Windows and Linux is detrimental to Microsoft's profit. As a consequence, the path cannot be optimal.

Some intuition with regard to the nature of this alternative path is of interest. Notice that for every cohort, the period profit function ($\pi(\rho)$) is a continuously differentiable, concave function of ρ that crosses ($\rho = 0$, $\pi = 0$), attains the unique maximum at $\frac{\beta(y)}{2}$, and crosses once again the point $\pi = 0$ at $\rho = \beta(y)$. Because Linux is present and an increase in q results in larger future profit, Microsoft will optimally set $p(t) < \frac{\beta(y(t))}{2}$ (this is the path characterized in the phase diagram above). However, because of the shape of $\pi(\rho)$, there is always a price $p(t) > \frac{\beta(y(t))}{2}$ that yields exactly the same level of *period* profit. Of course, if Microsoft was to set prices at this level in every period, $y(t)$ would fall until $y(t) = y^0$. This is the (non-profit maximizing) path that corresponds to this second case.

Proof of Proposition 3.

(a) According to equation (A10), the stable steady state y^{ss} satisfies

$$\beta'(y^{ss}(s)) = \beta(y^{ss}(s))\frac{r(s-1)}{s}$$

Differentiating implicitly with respect to *s*, we get

$$\frac{dy^{ss}(s)}{ds} = \frac{\beta(y^{ss})r}{s(\beta''(y^{ss})s - \beta'(y^{ss})r(s-1))} < 0$$

(b) Differentiate (A10) implicitly with respect to r to get

$$\frac{dy^{ss}(r)}{dr} = \frac{\beta(y^{ss})(s-1)}{\beta''(y^{ss})s - \beta'(y^{ss})r(s-1)} < 0$$

(c) That for all finite *s*, $y^{ss}(s) > y^0$ is obvious. Let $s \to \infty$. By equation (A10)

$\lim_{s\to\infty} \beta'(y^{ss}(s)) = \lim_{s\to\infty} \beta(y^{ss}(s))\frac{r(s-1)}{s}$ or $\beta'(\lim_{s\to\infty} y^{ss}(s)) = \beta(\lim_{s\to\infty} y^{ss}(s))r$

(because both β and β' are continuous functions of *s*). If $\lim_{s\to\infty} y^{ss}(s) = y^0$, it would have to be the case that $\beta'(y^0) = 0$, but this contradicts our assumptions. Furthermore, if $\lim_{s\to\infty} y^{ss}(s) = \infty$, it would have to be the case that $0 = \alpha_W r$, but this contradicts our assumption that $\alpha_W > 0$. We conclude that $y^0 < \lim_{s\to\infty} y^{ss}(s) < \infty$.

(d) By (A10), as $\to 1^+$, we have $\beta'(y^{ss}) \to 0$ But the only value of y for which $\beta'(y) = 0$ is y = ∞. Therefore, it must be that $y^{ss} \to \infty$.

(e) As $r \to \infty$, $\frac{r(s-1)}{s} \to .\infty$. For $\beta'(y^{ss}(s)) = \beta(y^{ss}(s))\frac{r(s-1)}{s}$ to hold, $\beta(y^{ss})$must be sufficiently close to zero. But $\beta(y^{ss})$ only gets arbitrarily close to zero when y is close to y^0.

(f) See the proof for (d).

Proof of Corollary. Using (A1), (A9), and (A10), we solve for the steady-state price, quantity, and profit:

$$p^{ss} = \frac{\beta(y^{ss})}{s+1} \qquad q^{ss} = \frac{s}{s+1} \qquad \text{and} \qquad \pi^{ss} = \frac{\beta(y^{ss})s}{(s+1)^2}$$

Notice, finally, that for a duopoly to happen in the steady state, it must be the case that $s > 1$. Comparing the expressions above to the case of a Windows monopoly proves the corollary.

Proof of Proposition 6. Solving Microsoft's problem as in the proof of Proposition 2, we obtain the equation that characterizes the saddle point:

$$\frac{\beta'(y^{ss})}{\beta(y^{ss})}\frac{s}{r} = \frac{s-1+\varepsilon(1+s)}{1-\varepsilon(1+s)} \tag{A11}$$

Notice, that the right-hand side is positive for all $\varepsilon \in [0, \frac{1}{1+s})$ and negative for $\varepsilon \in (\frac{1}{1+s}, 1]$.

Clearly, when the right-hand side of (A11) is negative, there is no y^{ss} (with $\beta(y^{ss}) > 0$, so that Windows is a player) that satisfies the equation.

Also, rewriting (A11) as

$$\beta'(y^{ss})\frac{s}{r}\frac{1-\varepsilon(1+s)}{s-1+\varepsilon(1+s)} = \beta(y^{ss})$$

and using the fact that $\lim_{\varepsilon \to \frac{1}{1+s}} \frac{1-\varepsilon(1+s)}{s-1+\varepsilon(1+s)} = 0$, we see that $y^{ss} \to y^0$ as $\varepsilon \to \frac{1}{1+s}$ (or as $\frac{1}{1+s} \to \varepsilon$). Thus, as ε grows (for given s) or s grows (for given ε), Windows is effectively pushed out by Linux.

Appendix C: Governmental Sponsorship of Linux

	Linux camp	MS camp	Both Linux and MS	OSS policy pending or in place	Policy exclusively OSS	Policy encourages OSS on case basis	Emotional / philosophical / nationalistic support for OSS	Official gov't organization to promote OSS	Notable projects	Institutions on Linux	Important factors
Argentina	x			x	x				Ututo	Department of Environmental Policy, Secretaría de la Gobernación, the Central Police Department, the Catholic University	
Australia	x			none, but considering it							
Austria		x		none	none	none	none	none		Green Party, primary schools	
Belgium	x			x	x					Computer Center of the Region of Brussels, Belgian National Army, Royal Botanic Garden	
Brazil	x			x	x						
Canada				none							
China			x	x			x			Ministry of Education, China Post Office, Hong Kong government	Piracy, avoid dependency on MS
Colombia				x							
Denmark		x		none	none	none	none	none		All schools	
European Commission	x			x	x		x	IDA?	eEurope		
Finland	x			x (not legislation)	x (not legislation)				gov't-led Linux seminars for employees		
France	x			x	x		x		Agency for Technologies of Information and Communication in Administration (ATICA)	Culture, Defense, Justice, Finance, and Education Ministries; National Crime Register; universities;	
Germany	x			x		x			Dortmund e-gov't	Federal Institute of Agriculture and Food, Administration of the German Parliament, Council of the Eldest, Police in Lower Saxony, Ministry for Inner Affairs, Municipality of the city of Munich, Institute for Livestock Breeding	security (avoid monoculture of software), cost savings
Ghana		x		none	none	none	none	none	telephone billing system		
Goa									Goa Schools Computer Project (Red Hat)		
India			x	x		x					
Italy				x	x		x				
Japan			x	x							
Kenya				x							
Korea	x			not sure							
Malaysia	x			x				Malaysian National Computer Confederation's Open Source Special Interest Group's (MNCC-OSSIG)			
Mexico			x	none	none	none	none		Red Escolar - botched implementation in schools	all areas of capital administration	
Nambia	x								SchoolNet Nambia		
NATO		x		none	none	none	none				
Nepal				not sure					school system project		
Netherlands			x	action plan							
New Zealand				none, but considers OS as option					Housing New Zealand, OpenLDAP		
Nigeria		?		no info							
Norway	x			none							
Pakistan	x			unclear		x		Technology Resource Mobilization Unit	e-governance, rural schools		
Peru			x	x	x						
Philippines			x	none yet, though may move towards it							
Poland				none yet					Hotel Jan Sobieski III	Polish Customs	
Portugal	x			x	x						
Russia		x									
Senegal				x							
Singapore	?			none, but considering gov't usage							
South Africa			x	x		x	x	Government Information Technology Officers Council (GITOC) working group on Open Source			
Spain				proposed					Extramadura school district	Senate, Nuclear Security Council, Ministry of Home Affairs, Ministry of Justice, Ministry of Public Administration (MAP project)	
Sweden			x	x		x					
Switzerland		x									
Taiwan			x	national plan							
Thailand	x			support, but no policy					SchoolNet		
Uganda	vibrant community			none							
UK			x	x		x				National Health Service, e-Government	
US			x	TX, OR, OK, CA	x	x				departments of Agriculture, Education, Energy, Health and Human Services, Housing and Urban Development, Labor, Treasury and Veterans Affairs, Navy, Air Force, the Federal Aviation Administration and the Defense Department, Department of Transportation, D	
Venezuela				x	x		x		banks		

References

Argentesi, Elena, Matteo Alvisi, and Emanuela Carbonara. "Piracy and Quality Choice in Monopolistic Markets." SSRN Electronic Paper Collection, no. 341960 (September 2002).

Arrow, Kenneth J. "Optimal Capital Policy, the Cost of Capital and Myopic Decision Rules," *Annals of the Institute of Statistics and Mathematics*, Tokyo, Vol. 16, pp. 21-30 (1964).

Bessen, Jim. "Open Source Software: Free Provision of a Complex Public Good." Open Source Research Community – MIT (2002).

Bitzer, Jürgen, and Philipp J. H. Schröder. "Bug-Fixing and Code-Writing: The Private Provision of Open Source Software." Discussion Papers of DIW Berlin from, German Institute for Economic Research, no. 296 (September 2002).

Dalle, Jean-Michel, and Nicolas Jullien. "'Libre' Software: Turning Fads into Institutions?" Working Paper, forthcoming in *Research Policy* (2002).

David, Paul. "CLIO and the Economics of QWERTY." *American Economic Review: Papers and Proceedings* 75 (1985): 332-337.

Dolley, Andrew. "IBM Claims SCO Conspiring with Microsoft over Linux," ZDNet Australia, July 30, 2003 (at http://www.zdnet.co.nz/newstech/enterprise/story, accessed on August 6, 2003).

Farrell, Joseph and Garth Saloner. "Standardization, Compatibility, and Innovation." *Rand Journal of Economics* 16 (1985): 70-83.

Farrell, Joseph and Garth Saloner. "Installed Base and Compatibility: Innovation, Product Preannouncements, and Predation." *American Economic Review* 76 (1986): 940-955.

Foster, Richard. *Innovation : The attacker's advantage*. New York: Summit Books (1986).

Fuller, Thomas. "How Microsoft Warded off Rival," *New York Times*, May 15, 2003.

Johnson, Justin Pappas. "Open Source Software: Private Provision of a Public Good." *Journal of Economics and Management Strategy*, 11, no. 4 (2002): 637-62.

Kamien, Morton and Nancy Schwartz. *Dynamic Optimization*. Elsevier Health Sciences; 4th reprint, 2000 edition (October 1, 1991).

Kogut, Bruce, and Anca Metiu. "Open-Source Software Development and Distributed Innovation." *Oxford Review of Economic Policy*, 17, no. 2 (2001): 248-64.

Kuan, Jenny. "Understanding Open Source Software: A Nonprofit Competitive Threat." Working Paper (September 23, 1999).

Lakhani, Karim R., and Eric von Hippel. "How Open Source Software Works: 'Free User-to-User Assistance.'"MIT Sloan School of Management Working Paper, # 4117 forthcoming in *Research Policy* (June 2002).

Lerner, Josh, and Jean Tirole. "Some Simple Economics of Open Source." *Journal of Industrial Economics* 50, no. 2 (2002): 197-234.

Mockus, Audris, Roy T. Fielding, and James D. Herbsleb. "A Case Study of Open Source Software Development: The Apache Server." Paper presented at the ICSE 2000, Proceedings of the 22nd International Conference on Software Engineering, Limerick, Ireland, June 4-11, 2000.

O'Mahony, Siobhán. "Guarding the Commons: How Community Managed Software Projects Protect Their Work." Stanford Working Paper Collection, forthcoming, 2003, *Research Policy* 2002.

Ross, David R. "Learning to Dominate." *The Journal of Industrial Economics*, Vol. 34, No. 4. (Jun., 1986), pp. 337-353.

Schmidt, Klaus M., and Monika Schnitzer. "Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market." SSRN Electronic Paper Collection (July 12, 2002): 37.

Seierstad, Atle and K. Sydsæter. *Optimal Control Theory with Economic Applications*. North-Holland; 3rd reprint 2002 edition (1st edition: February 1, 1987).

Silverman, Brian. "Sun Microsystems, Inc.: Solaris Strategy," HBS case 9-701-058. February 2, 2001.

Spence, A. Michael. "The Learning Curve and Competition." *The Bell Journal of Economics*, Vol. 12, No. 1. (Spring, 1981), pp. 49-70.

Thorn, Brian K., and Connolly, Terry. "Discretionary Data Bases: A Theory and Some Experimental Findings." *Communication Research*, Vol 4.5, October, 1987, 512-528.

Xu, Xiaopeng. "Development Costs and Open Source Software." Working Paper (2002).