Open Source Standardization: The Rise of Linux in the Network Era

Joel West
<joelwest@ieee.org>

and

Jason Dedrick
<jdedrick@uci.edu>

Center for Research on Information Technologies and Organizations (CRITO)
University of California, Irvine
3200 Berkeley Place
Irvine, CA 92697-4650
+1-949-824-2863; fax: +1-949-824-8091
http://www.crito.uci.edu

November 1, 2001

Contents

To appear in *Knowledge, Technology & Policy*
Special issue on “Information Technology Standards, Standardisation and Policies”
Vol. 14, No. 2 (Summer 2001)

This paper is updated and expanded from
“Proprietary vs. Open Standards in the Network Era: An Examination of the Linux Phenomenon,”
Proceedings of the 34th Hawai‘i International Conference on System Sciences, Jan. 2001.

Open Source Standardization:
The Rise of Linux in the Network Era

Abstract: To attract complementary assets, firms that sponsor proprietary de facto *compatibility standards must trade off control of the standard against the imperative for adoption. For example, Microsoft and Intel in turn gained pervasive adoption of their technologies by appropriating only a single layer of the standards architecture and encouraging competition in other layers.*

In reaction to such proprietary strategies, the open source movement relinquished control to maximize adoption. To illustrate this, we examine the rise of the Linux operating system from 1995-2001, particularly the motivations of organizational buyers and suppliers of complementary assets, and Microsoft's reaction to its success..

I. Introduction

Vertically integrated proprietary *de facto* standards architectures were the norm for the first three decades of the postwar computer industry. Each computer maker developed most if not all of its technology internally, and sold that technology only as part of an integrated computer system. This systems era was ascendant from IBM's 1964 introduction of its System 360 until the 1981 release of IBM's first personal computer (Moschella 1997).

This strategy was challenged by two different approaches. One was the fragmentation of proprietary standards in the PC industry between different suppliers, which led firms like Microsoft and Intel to seek industrywide dominance for their proprietary component of the overall system architecture, marking what Moschella (1997) terms the "PC era" (1964-1981). The second was a movement by users and second-tier producers to create industrywide "open" systems, in which the standard was not owned by a single firm.

The explosive adoption of the Linux operating system in the late 1990s was a response to these earlier approaches. Linux was the most commercially successful example of a new wave of "open source" software, in which the software and even source code are freely distributed to use and modify. At the same time, its adherents emphasized its advantages in contrast to the proprietary PC standards, particularly software standards controlled by Microsoft.

In the next section, we examine the evolution of standards competition strategies in the computer industry from the proprietary integrated systems to both horizontal specialization and open systems. We then trace the birth and growth of Linux and related Open Source systems, and then review rival explanations for their recent successes (Table 1). Finally, we conclude with an analysis of the durability of the Linux phenomenon in the network era.

II. Era of Proprietary Standards Competition

Product compatibility standards have typically been considered using a simple unidimensional typology, bifurcated between “compatible” and “incompatible.” However, to illuminate differences between proprietary and open standards strategies, we use Gabel’s (1987) multi-dimensional classification attribute, with each dimension assuming one of several (discrete) levels:

- “multivintage” compatibility between successive generations of a product:
- “product-line” compatibility, providing interoperability across the breadth of the company’s product line — as Microsoft has with its Windows CE, 95/98/ME, and NT/2000 product families.
- “multivendor” compatibility, i.e. compatibility of products between competing producers.

Vertically Integrated Proprietary Systems

During the era of vertically integrated computer systems, the most successful architecture was IBM’s System 360 (later 370 and 390) line of mainframe computers. The 360 marked IBM’s switch from being a major buyer of electronic components to one of the largest manufacturers, all for internal use (Sobel 1981). IBM produced most of the hardware — including electronic components, CPU board and peripherals — as well as the operating system, tools and much of the application software. Outside IBM, its mainframe and minicomputer competitors from 1950-1980 also sought vertical integration, although most lacked IBM's scope to achieve the same level of integration.[1]

IBM and its System 360 were fantastically successful in this strategy, at one point garnering nearing half of the computer industry's profits (Moschella 1997). A key reason was that the System 360 marked the first widespread implementation of what Gabel (1987) refers to as "product line compatibility." Unlike previous generations, IBM shared the same standards architecture across its product line — and thus enabling use of the same operating system, tools and user applications (Figure 1).

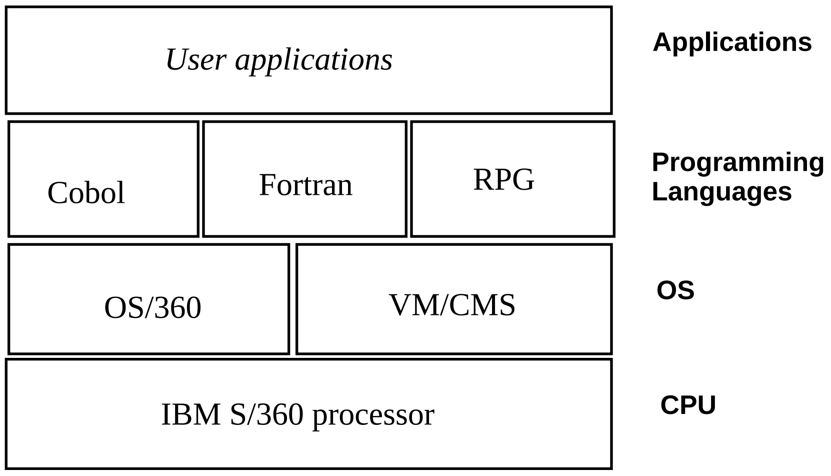


Figure 1: IBM System 360 architectural layers

But IBM was forced to limit its vertical integration in the face of a 1969 federal anti-trust lawsuit. To respond to one of the charges, that of illegal "tying" of its mainframe hardware and software, five months later IBM decided to end its decades-long practice of bundling its offerings, allowing companies to buy the hardware, software and services separately. As hoped by the government, this unbundling encouraged the third-party supply of peripherals, software and support; as one vendor put it, "when IBM says there is such a concept as buying software separately, it makes it acceptable to customers" (Sobel 1981).

Still, many IBM customers bought a complete IBM solution. This helped IBM keep customers, because those customers who had custom-developed software would find it more expensive to adapt their

[1] Nor was such vertical integration unique to computers, as it was the approach used by AT&T until its 1984 break-up. Moschella (1997) notes that similar vertical integration could also be found in the early states of other technology-based industries, such as the U.S. radio and television broadcasting industries.

software to a new processor, operating system or compiler.[2] As Greenstein's (1997) study of U.S. government computer purchases showed, such strategies helped discourage switching between vendors. The largest minicomputer vendor, Digital Equipment Corporation, used a similar strategies in the 1970s and 1980s to retain customers, as when it enhanced programming language compilers with extensions not available on competitors' machines.

At the same time, IBM's vertical integration strategy had its limits as an exemplar for the rest of the industry. Without IBM's economies of scale and scope, rivals were eventually forced to choose between inferior performance on key dimensions, obtaining key components from outside suppliers, or (as GE and RCA soon did) exiting the computer business.

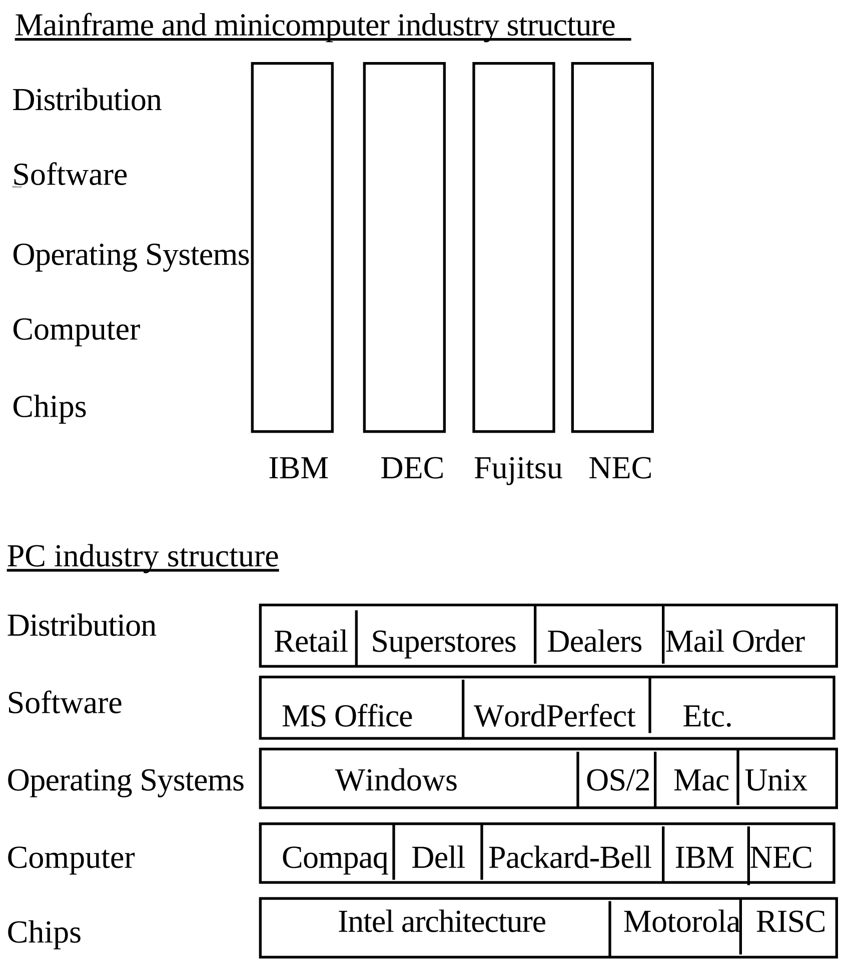
Horizontal Specialization and Quasi-Monopolies

The first serious attack on the fully-integrated proprietary systems was against the integration rather than the proprietary nature. Ironically, IBM itself enabled this switch when, its haste to ship a personal computer, it purchased the processor and OS components from outside vendors. Its respective contracts with Intel and Microsoft enabled these firms to sell their components to rival systems vendors, enabling the IBM PC-compatible computer industry (Chposky and Leonsis 1988; Moschella 1997).

The net result was the personal computer industry developed with a standards environment that emphasized horizontal specialization in a given layer of the system, rather than vertical integration of internally developed components. In particular, each supplier had an incentive to obtain an industry-wide quasi-monopoly by selling its hardware or software to as many systems vendors as possible (Figure 2). This interpretation of the industry transformation is often attributed to the work of Grove (1996).

Grove (1996) presents it as a normative rule in the strongest possible terms, arguing firms that failed to recognize the transformation risk being swept away. By the mid-1990s it was clear that the horizontal proprietary model had overwhelmed the vertical one, at least as measured by overall industry spending.

[2] Some CPUs were compatible with IBM's S/360 line, notably Amdahl (later Fujitsu) and NAS (later Hitachi). However, these apparently constituted a small fraction of IBM's sales, at least through the late 1970s (Greenstein 1997).



Source: Adapted from Grove (1996)

Figure 2: Proprietary standards strategies in the computer industry

Competitive Advantage through Proprietary Standards

Both the vertically integrated and horizontally segmented strategies used in the computer industry assumed proprietary (single-vendor) control of one or more standards in a systems architecture. The differences between these proprietary approaches (and with subsequent "open" standards) highlight the conflict between a firm's competing objectives in standards competition: adoption and appropriability.

Widespread adoption of the standard is important because the CPU, operating system and key application tools such as compilers and databases are more widely valued by users if they have a wide range of compatible software. Thus, to get the widest variety of software built upon their respective layer of the architecture, sponsors of a standard use pricing and other incentives to gain the largest number of early users and encourage the development of co-specialized software (Teece 1986; Morris and Ferguson 1993; Shapiro and Varian 1998).

At the same time, success of the standard provides no guarantee as to the sponsor's ability to profit from that standard. That ability will depend on intellectual property protection for the standard, and the relative importance of standard and complementary assets (Teece 1986). Control over the architecture enables the technological evolution of the standard and provides the incentive for the necessary investments (Morris and Ferguson 1993). However, in the increasingly common case of divided control of a standards architectures, control of application programming interfaces (APIs) determines access to the software and thus the right to profit from a standard (West and Dedrick 2000).

So the sponsors of successful standards face competing incentives, as illustrated by the 1980s experience of the PC industry. They can gain wide adoption of their standard, at the risk of not profiting from its success (as IBM did). Or they can control the profit from the standard, at the risk of limited adoption and potential abandonment (as Apple did). If they try a compromise of both adoption and control — as Sun did with European standards bodies and its Java programming language (Egyedi 2001) — they can easily end up with neither.

The horizontal model adopted by Microsoft and Intel is an exemplar of what Morris and Ferguson (1993) termed "open but proprietary" standards — surrendering the appropriability of vertical integration in hopes of adoption as a universal layer in an industry-wide standards architecture. Of course, as both Morris and Ferguson (1993) and Liebowitz and Margolis (1999) remind us, success also requires successful delivery of product features and quality.

III. Unix Enables Open Systems and Networked Computing

Unix and its Bicoastal Childhood

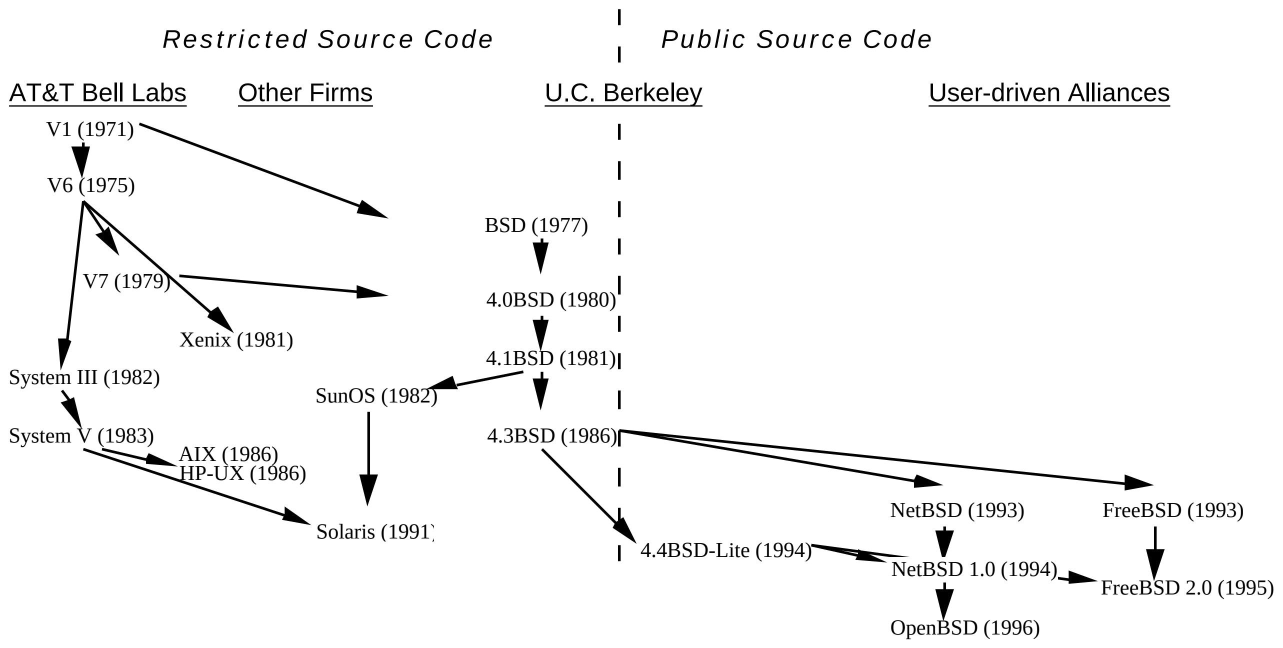
The first successful multi-vendor operating system was Unix, developed by a computer science research group at Bell Telephone Laboratories (BTL) in New Jersey beginning in 1969. As with the earlier Multics research project between MIT, BTL and mainframe computer maker General Electric, Unix was a multi-user time-shared operating system designed as a research project by programmers for their own use. At the same time, it included hallmark innovations such as modular commands and online programmer documentation.

Other characteristics key to Unix's success reflected path dependencies by its developers and early users (Salus 1994):

- AT&T was forbidden by its 1956 consent decree from being in the computer business, so it did not sell the OS commercially. But after publishing influential research papers (e.g. Ritchie and Thompson 1974), Bell Labs was flooded with requests from university computer science departments, who received user licenses and source code but no support.
- Because of budget constraints that limited BTL researchers to DEC minicomputers rather than larger mainframes, Unix was simpler and more efficient than its Multics predecessor, based on the simplified C programming language rather than the more unwieldy PL/I.
- Although originally developed for DEC minicomputers, Unix was converted to run on other models by users who found programmer time less expensive than buying a supported model, thus setting the stage for it to become a hardware-independent operating system.

But perhaps the most important development was the licensing of Unix by the U.C. Berkeley Computer Science Department in 1973. The Berkeley group issued its own releases from 1977 to 1994, with much of its funding provided by the Defense Advanced Research Projects Agency (DARPA). The result of the Berkeley development included (Garud and Kumaraswamy 1993; Salus 1994):

- the first Unix version to support TCP/IP, later the standard protocols of the Internet;
- academic adoption of BSD Unix as the preferred OS by many computer science departments throughout the world;
- commercial spread of BSD-derived Unix through Sun Microsystems, cofounded by former BSD programmer Bill Joy;
- as they evolved their respective versions of Unix, fragmentation of Unix developers and adopters into rival "BSD" and "AT&T" camps (Figure 3).



Sources: Salus (1994), Schneider (2000), company web sites

Figure 3: Simplified genealogy of Unix implementations

The Push for Formal Standardization: The "Open Systems" Movement

AT&T's Unix provided a multivendor standard which, when coupled with the BSD enhancements, helped spur the adoption of networked computing. Aided by Sun, whose slogan became "the network is the computer," Unix rapidly gained acceptance during the 1980s as the preferred operating system for networked engineering workstations (Garud and Kumaraswamy 1993). At the same time, it became a true multivendor standard as minicomputer producers with a small customer base, weak R&D and immature operating system licensed Unix from AT&T. The main exceptions to the Unix push were the early leaders in workstations (Apollo) and minicomputers (DEC), who used their proprietary operating systems as a source of competitive advantage, and thus were the last to switch to Unix in their respective segments.

Advocates among both producers and users formed a number of trade associations to promote Unix and related operating systems. By fueling the adoption and standardization of Unix, they hoped to increase the supply of application software to compete with sponsored, proprietary architectures (Gabel 1987; Grindley 1995). These two groups promoted these under the rubric "open systems"; the editors of a book series on such systems summarized their goals as follows:

> Open systems allow users to move their applications between systems easily; thus, purchasing decisions can be made on the basis of cost-performance ratio and vendor support, rather than on which systems will run a user's application suite (Salus 1994: v).

Despite such noble goals, the Unix community spent the 1980s and early 1990s fragmented into AT&T and Berkeley warring factions, each of which sought control of the OS APIs to maximize the software available for their respective versions. Each faction had its adherents. To avoid repeating earlier mainframe switching costs, U.S. Department of Defense procurement decisions began to favor Unix over proprietary systems. As AT&T formalized its System V Interface Definition and encouraged hardware makers to adopt System V, it became the multivendor standard required by DoD procurements.

Because BSD group developed only for DEC minicomputers, its Unix variant was not multivendor and thus less attractive for DoD procurements. However, the numerous innovations of the BSD group in terms of usability, software development tools and networking made it more attractive to university computer scientists for their own research and teaching, making it the minicomputer operating system preferred by computer science departments in the U.S., Europe and Japan (Salus 1994).

The divergent innovation meant that the two major Unix variants differed in terms of internal structure, user commands and application programming interfaces (APIs). It was the latter difference that most seriously affected computer buyers, as custom software developed for one type of Unix could not directly be recompiled on the other, adding switching costs between the two systems. In addition, both the modem-based and DARPA networking facilitated the distribution of user donated source code libraries, which were free but often required site-specific custom programming if the Unix APIs at the user's site differed from those of faced by the original contributor.[3]

[3] The idea of user-donated software had originated with SHARE, a volunteer group of IBM mainframe users that was founded in 1955 (SHARE 2001). However, SHARE's distribution of software during the 1960's and 1970's was organized around the physical exchange of magnetic tapes at periodic user meetings, which diffused the software less quickly or broadly that the interconnected networks that developed around Unix systems.

To reduce switching costs between Unix variants, in 1981 the IEEE formed a POSIX working group to develop a multivendor operating system standard.[4] The POSIX standard initially focused on defining a common set of APIs, so that software could be written that was source code compatible between multiple implementations. The standardization effort was supported by the U.S. government, which used POSIX compatibility as a requirement for procurement of midrange and later mainframe computers (Ambrosio 1987; Taft 1987). As the POSIX family of standards defined core APIs, and later enhancements and user tools, Unix vendors worked to evolve their own implementations towards POSIX compliance, both to satisfy customer procurement demands and also leverage the growing library of POSIX-compatible software.

As the independent POSIX specification was being developed, the “AT&T” and “BSD” factions joined forces with a 1987 agreement by AT&T and Sun (the leading BSD vendor) to merge their two streams into a common technical standard. Fearing AT&T control and increasing licensing fees, in 1988 larger computer makers established a rival group, the Open Software Foundation. But in 1989 both OSF and the AT&T-led Unix International effectively ended their rivalry by joining X/Open, a group of weaker European computer makers established in 1984.[5] The APIs published after negotiation among X/Open members then became the common Unix standard for developers and end users (Gabel 1987; Garud and Kumaraswamy 1993; Grindley 1995).

As the API divisions within the Unix community gradually ended, a new split developed based on intellectual property rights, specifically between those versions of the operating system based on code licensed from AT&T, and those which were independent of the AT&T sources. The licensed OS versions

4 POSIX stands for Portable Operating System Interface. The IEEE working group in charge of POSIX was later renamed the Portable Application Standards Committee (PASC). After the IEEE had established its first standard, a parallel POSIX group was established in Europe under the sponsorship of JTC1, which in turn is jointly sponsored by the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC)

5 In 1990, AT&T created a new subsidiary to handle all Unix-related marketing and development, and then exited Unix altogether in 1992 when it sold the subsidiary and all associated rights to Novell.

included all the early Berkeley releases, as well as most of the commercial versions of the software, including the Xenix operating system from Microsoft.[6]

At the same time, new university and hobbyist implementations of Unix-like operating systems were launched to avoid the restrictive AT&T licensing royalties and non-disclosure terms. This would allow free (or nearly free) software to be distributed in source code form to the widest possible audience. The POSIX specifications facilitated the independent development of a Unix-compatible OS without access to the AT&T sources. Among the earliest such OSs was Minix, explicitly developed for teaching computer science (Tanenbaum 1987). Meanwhile, the 4.4 BSD developers fought and eventually won the right to release (in source code form) a version of BSD which did not include AT&T restricted code, which in turn spawned the NetBSD, FreeBSD and OpenBSD operating systems. The volunteer groups developing these Unix clones were the forerunners of what later became the “Open Source” movement.

IV. GNU, Linux and Open Source

The ARPANET (and its successor, the Internet) provided a reliable way to organize dispersed groups of software engineers, through e-mail, discussion groups and primitive file servers. These groups worked to develop and expand the network, as well as the tools necessary to run it, such as e-mail and later web servers. Others worked on free Unix-like operating systems, eventually including Linux.

Software and Growth of Internet

The ARPANET had begun as a research network sponsored by the U.S. Department of Defense. So in 1986, government-sponsored researchers began quarterly meetings of what became the Internet Engineering Task Force. The group quickly broadened to include other U.S. and later international researchers, and in 1998 sponsorship shifted from the U.S. government to the non-profit Internet Society. While it lacks the statutory enforcement mechanisms of a formal *de jure* standards body, with implicit U.S. government sponsorships and a formal standardization process, the IETF standards are closer to *de*

[6] In addition to trade secrets and copyrights to Unix, AT&T (and its later successors) vigorously enforced their Unix trademark, forcing both producers and user groups to go to great lengths to associate their efforts with Unix without using the trademark (Salus 1994).

jure than *de facto* standards. At the same time, structurally the IETF is an *ad hoc* organization subdivided into working groups that focus on the design and implementation of the standards used to run the network, as well as key operational issues for running an exponentially growing communications system.

Its process is based on “rough consensus and running code.” The adoption of any standard in final form requires at least two independent and mutually compatible implementations of that standard; when possible, the standard is defined in terms of a “reference implementation” with public source code. Unlike some standards organizations, the IETF discussions, draft specifications and all source code are public documents posted on Internet file servers (Malkin 1993; Bradner 1999). As Bradner (1999: 52) noted “The IETF supported the concept of open sources long before the Open Source movement was formed.”

These prototype implementations were increasingly developed for use on Unix-compatible computer systems. As the growth of the Internet shifted the user base from workstations to primarily personal computers, the client-based Unix implementations declined in importance, but they continued to enable the use of Unix (and the Unix clones) to run Internet servers using well-tested free software such as Sendmail, BIND, and Apache.

GNU Becomes Linux

The origin of the open source movement came in 1984, when MIT programmer Richard Stallman quit his job and began the GNU project. His was driven by opposition to the norms of commercial software: proprietary technologies, secret source code and software copyright. In his view, all software should be “free software,” with source code that can be read, modified and redistributed (Stallman 1999).

His GNU system began with the Emacs text editor and other development tools, and was widely distributed via tape and via the ARPANET research network and later the Internet. The various tools were designed to work on a POSIX-compliant operating system, with the GNU C and later C++ compilers especially popular with developers seeking to implement a quick programming language. However, GNU lacked the essential core of a modern operating system, a kernel. Efforts to develop a kernel had first been delayed by work on other components, and then floundered for several years without strong leadership.

The eventual kernel came from Linus Torvalds, a Finnish undergraduate student, who in August 1991 announced in a worldwide Minix discussion group his plans for a free hobbyist operating system for Intel-based PCs. With Minix as a development platform and the GNU tools providing essential functionality, Torvalds used the Internet to develop the source code in collaboration with a loose federation of dozens of programmers. By the end of 1992 they had implemented Linux, a POSIX-compliant operating system which was distributed free via a collection of FTP servers scattered around the world (Klaus 1993).

"Open Source" Movement

By the mid-1990s, examples of "free software" included Project GNU, Linux, and the various BSD-derived OS and tools. But different groups argued over what freedoms were necessary and desirable. All agreed that users should have the freedom to read, use and modify their source code, but differed on two key issues: organization and appropriability (Table 2).

Type of Software	Zero Price	Source Code Available	Source Code Modifiable	Public changes to core code	All derivatives must be free
Commercial					
Shareware	†				
Open Source (BSD-Style)	X	X	X		
Open Source (Apache Style)	X	X	X	X	
Open Source (Linux/GNU style)	X	X	X	X	X

† Nominal price, but unenforced

Source: Abridged from Valloppillil (1998)

Table 2: Comparison of commercial and "open source" software licenses

GNU and the free versions of BSD use a volunteer analog to the organizational approach used for proprietary commercial software, with development is done by a small and carefully coordinated group. The Linux development is largely without structure and accepts contributions from anyone, with marketplace feedback from thousands of adopters to determine which changes were kept and which ones would be rejected. Raymond (1999) respectively labeled these two models "the cathedral and the bazaar."

More significantly, some developers vehemently object to the idea that free source software be merged with proprietary software or modified to become proprietary; therefore, source distributed under the GNU Public License forbids such modifications. A rival group worried that the anti-capitalist rhetoric (particularly by Stallman) would limit the adoption of free software by both developers and users. They drafted a rival license, the "Open Source Definition," which allowed for commercialization. This business model was quickly endorsed by Torvalds, Netscape, and other key actors (OpenSource.org 1999).

In response to the increasing popularity of Open Source and Linux, a number of firms offered partial Open Source strategies to spur adoption and thus complementary assets:

- In 1998, Netscape released the source to its browser to the open source group Mozilla.org, although outsider contributions were dwarfed by those from Netscape employees.
- Later that year, Sun announced the distribution (under a “Sun Community Source License”) of Java-related source code but not its Solaris operating system.
- In 1999, Apple announced its “Darwin” open source initiative, providing the BSD part of its new Mac OS X operating system but excluding Macintosh emulation and its new Aqua user interface.

Interestingly, even the most restrictive “free software” license does not require that the software be distributed without charge: “to understand the concept, you should think of ‘free speech,’ not ‘free beer’” (Free Software Foundation 2000). From a practical standpoint, the earliest GNU distributions charged a nominal handling fee for mailing magnetic tapes; more recent free software distributions have been sold on CD-ROM, even though all contemporary open source projects are freely available on the world wide web. This intentional loophole enabled creation of for-profit distribution firms such as SuSE (1992), TurboLinux (1992), VA Linux (1993), Red Hat (1994) and Caldera Systems (1998).

Commercialization of Linux

The notion of commercializing Linux seemed a contradiction to many. Commercial software development is based on the premise that the developer must be able to protect the intellectual property embedded in software code through trade secrets, copyright and even patents. Without intellectual property protection, there was no way for the developer to collect economic rents and thus no incentive to develop new technologies. Simply put, how can someone make money selling something that was given away for free on the Internet?

The answer turned out to be simple: the same way that companies sell bottled drinking water when it’s available for virtually free out of the tap. Just as bottled water companies filter drinking water so it tastes better, new companies started to package Linux with supporting applications, documentation and services to add value and make it easier to use. Companies such as Red Hat offered “distributions” of

Linux on CD-ROM with installation software, utilities, and user manuals. They also usually offered some level of technical support, while others such as Linuxcare provided support for all Linux distributions.

After years of being developed by a group of reputedly idealistic programmers who selflessly gave their time without financial gain, Linux suddenly got caught in the updraft of the IPO frenzy of the late 1990s. Red Hat went public at $14 on August 11, 1999, and topped $250 by December of that year. By the end of its first day of trading, VA Linux shares had gained 698%, a new record for an IPO (Kawamoto and Shankland 1999). The IPOs filled the coffers of the new companies, some of whom rewarded the Linux community by offering pre-IPO shares to Linux programmers (although there was some resentment from other programmers who were left out).

The Linux boom was short-lived as stock prices collapsed in the spring of 2000. By the end of 2000, shares of Caldera, Red Hat and VA Linux had each lost 93-97% of their peak market value. The bursting of the stock bubble raised the issue of whether the Linux companies had developed viable profit models. However, it did not appear to slow the adoption of Linux, which already had established itself with a significant number of users, and was supported by major hardware, software and IT services companies. This continued growth raises the question of why these users and providers of complementary assets were now making Linux a key part of their technology strategies.

V. Adoption Motivations

End user adoption

Starting from negligible sales in 1995, Linux became the fastest growing operating system on the market, capturing one fourth of the server market in only four years. There are several possible explanations for the success of Linux against powerful entrenched competitors.

Fragmented Server Market. The operating system market for client PCs has evolved along the lines predicted by theories of increasing returns and network externalities (Shapiro and Varian 1999). Most PC users have strong incentives to choose Windows, given the huge library of Windows applications and the ability to share files with other Windows users. The result is a 92% market share for Windows in desktop and notebook PCs, as compared to 4% for the Mac OS and about 1% for Linux (Shankland 2001).

On the other hand, the server market was historically different. The market was once dominated by Novell Netware and various versions of Unix from Sun, HP, IBM, Compaq and SGI, but Windows gradually passed both. Still, in 2000 Windows held only 41%, and its share grew only 6% over the preceding three years, as compared to a 20% gain in share by Linux (Table 4).

	1995	1996	1997	1998	1999	2000
Windows NT/2000	18%	26%	35%	38%	38%	41%
Linux	0%	7%	7%	16%	25%	27%
Unix	25%	20%	21%	19%	17%	14%
Netware	35%	32%	27%	23%	19%	17%
Other	22%	16%	10%	4%	1%	1%
Total units	2.2m	3.1m	3.5m	4.4m	5.4m	6.1m

Source: International Data Corp., as reported by MacCormack and Herman (1999); Shankland (2001)

Table 4: Server operating system market share (global unit shipments), 1995-2000

It also is quite common for large enterprises to support more than one server platform. For instance, Dell Computer has run many applications on its own PC servers running Windows NT or Windows 2000, yet its order management system runs on high-end Tandem servers (Kraemer *et al.* 2000). In such a mixed environment, there may be less resistance to adopting another operating system to handle new functions such as web hosting.

New Uses. IT departments tend to resist changing platforms for established applications, as there are rarely compelling enough benefits to justify the high switching costs (and the threat to personnel whose expertise is in the existing platform). As a result, new platforms usually gain acceptance when applied to new uses. For instance, the minicomputer initially was adopted as a single-purpose platform for engineering applications, rather than a replacement for the general purpose mainframe computer. Likewise, the PC was used for personal productivity applications that were difficult to perform in a time-sharing centralized computing environment.

The most common applications for Linux are infrastructure applications, mostly centered around the Internet, where Linux servers are commonly used for web/Internet applications, e-mail, firewall, and proxy/caching/security services. According to IDC, 40% of all customer spending for Linux servers in 1999 was for Internet-related applications (Bailey *et al.* 2000). The fact that these applications were

Internet-centric made them ideal for Linux, which grew up on the Internet and was developed to a large extent with Internet applications in mind, just as Windows was developed to run desktop PC applications. Similarly, an estimated 60% of the web servers on the Internet used the open source Apache web server, which was created on and remains optimized for Unix-compatible operating systems (Lohr 2000).

Open source software such as Linux, BSD Unix or Apache has one other inherent attraction for Internet applications: user control. As bugs or security holes were discovered, a skilled Internet service provider could apply known fixes contributed by other users or develop its own fixes, without waiting for the software update cycle of the software vendor. Such control and predictability was particularly important for Internet servers which, unlike most office computers, had to work reliably 24 hours/day.

New Users. New standards are adopted not only for new applications, but also often by a new set of users who are not invested in existing technologies. The minicomputer was first adopted by scientists and engineers, rather than the data processing departments who were more locked into the mainframe, and PCs were adopted by individual users rather than companies or departments.

The early adopters of Linux were neither corporate IT departments, who were busy supporting various proprietary server platforms, nor individual PC users, who were using Windows. Instead, they included computer scientists, hackers, students, and others who wanted to be able to modify the software to fit particular uses, and who in many cases were part of the Linux programming community. They also included Internet service providers, web site developers, and intranet developers who wanted a low-cost, robust operating system suited to such Internet applications.

According to IDC, Linux is used in a variety of industries, with the highest concentration in educational institutions and the IT industry (Bailey *et al.* 2000). This is not surprising, given the high numbers of computer scientists and engineers in those two settings.

Also, as Lerner and Tirole (2000) point out, the greatest use of open source software such as Linux takes place “in settings where the end users are sophisticated...In these cases, users are apparently more willing to tolerate the lack of detailed documentation or easy-to-understand user interfaces in exchange for the cost saving and the possibility of modifying the source code themselves.”

Price/Performance. While Linux is a multi-platform OS like Unix, most Linux users have chosen to use low cost Intel-based hardware, which enjoys huge economies of scale from the global PC industry. Adopters say that Linux has better reliability than other OSes available on the Intel platform, such as Windows NT and NetWare. For example Burlington Coat Factory made a major commitment to Linux in 1998, buying 1,250 desktop PCs running Linux to manage the day-to-day retail business of its 264 stores. In the words of Mike Prince, Burlington's chief technology officer, "Linux is rock-solid. It has a tremendous amount of mind share. It is unifying Unix in a way that it has never been before" (New York Times 1999). This weakness of NT was directly acknowledged by Microsoft, which promoted its Windows 2000 server software as being "three times" more reliable than Windows NT.

Other organizations adopted Linux for managing large databases. Retailers such as Home Depot were attracted by the lack of a per-terminal license fee. Meanwhile, Shell Oil had switched to a Linux-based IBM mainframes to analyze oil exploration seismic data (Bulkeley 2001).

Adoption by software and service vendors

A key issue in any standards competition is whether a standard can attract developers of complementary assets, which account for much of the value of a standards platform (Teece 1986). In order to succeed Linux needed a supply of applications, development tools, and support services, especially if it was to be accepted in more conservative business settings.

The first key complementary asset for Linux originally was the large pool of professionals who wrote most of the code, found and fixed bugs, and also provided technical support to users via the Internet. In many cases, a user could get a problem diagnosed and solved via e-mail faster than he or she would get similar support from a software vendor's technical support team. For businesses and other large institutions, however, relying on the kindness of strangers is simply too risky. MIS departments are used to having support contracts with vendors or third party service providers, and having legal recourse if such support fails them. This required that someone step in and offer service and support on a commercial basis if Linux were to penetrate deeper into user organizations.

Just as the Linux community provided valuable support for users, the open-source community was also responsible for the first wave of applications to run on Linux, including Apache web server, Sendmail e-mail routing, Perl and TCL programming languages, and the MySQL database. To advance beyond running Internet servers, however, Linux needed software companies to port leading business applications to the Linux platform.

As Linux evolved, it began to attract support from commercial suppliers of services and software. First, were the startups such as VA Linux, Red Hat, Caldera, SuSE, and TurboLinux. Some of these also offered consulting services, training and certification, and technical support on a contract basis. Others developed applications for Linux, including Linux distributors such as Red Hat and SuSE and independent software vendors such as Applixware (Gillen and Kusnetzky 2000).

Around 1998, a new wave of companies entered the picture. These were large established software and services vendors, such as IBM, Oracle, Novell, Hewlett-Packard, Computer Associates, and Corel. They began to port applications to Linux, from productivity suites such as Sun's Star Office and Corel's WordPerfect to enterprise applications such as Oracle databases and IBM's DB2 database application and MQ Series messaging software. The support of these established suppliers was vital to providing confidence to potential users that Linux was a viable standards choice for business applications.

Why did these providers of complementary assets choose to support Linux, when they had ignored other standards? The decision was not without costs or risks, especially to more established vendors. Some had their own operating systems that might be threatened by the success of Linux (e.g., Compaq, HP and IBM, which had their own versions of Unix as well as other proprietary OSes). Second, there were significant costs to developing and supporting Linux versions of major applications, and the Linux market was still small and growth uncertain. Third, Linux did not offer the opportunity to lock customers in to the vendors' own technologies. Finally, there was the threat of retaliation from Microsoft, which held significant leverage over many of these companies.

Under these conditions, such widespread support for Linux might be explained by the following factors:

- Responding to customer demand: As Linux gained traction in organizations, there was a need for more robust support, and users turned to their traditional IT vendors. According to IBM, its decision to offer Linux on the System 390 mainframe was made “in response to a groundswell of demand by S/390 users” (IBM 2000).

- Cost: Vendors such as IBM, HP and Compaq may be hoping to reduce their own software development costs, and are looking at Linux as an alternative to their second-tier Unix systems. In this case, Linux is ideal, as its development costs are borne by the Linux community, and adopting it does not put the vendors at the mercy of a competitor.
- Strategic reasons: Companies such as Oracle, Sun, and IBM all compete aggressively with Microsoft, and see Linux as a tool for challenging Microsoft in the server market. IBM and HP also are anxious to catch up to Sun in fast-growing hardware markets for Internet and e-commerce servers, and are capitalizing on the popularity of Linux in the Internet world.

Adoption by hardware vendors

PC Vendors. Linux originally was developed on the Intel platform, and is still predominantly used on Intel servers. Early adopters used older PCs, self-built systems, or so-called “white box” clones. As of September 1999, self-built systems accounted for 39.5% of installed Linux servers and white boxes another 14.9%, according to IDC. However, the big PC makers have entered the market and captured a growing share, with Compaq, IBM, HP and Dell leading the way.

Why did PC companies offer Linux as an alternative to Windows NT and later Windows 2000?

- Customer demand: As in the case of software and service providers, hardware companies are responding to demand for Linux. If end users are going to download Linux or install it from a CD-ROM, PC makers would rather offer it pre-loaded and give customers what they want on the vendor’s own hardware.
- Independence from Microsoft: PC makers have been entirely dependent on Microsoft for operating systems for both client and server PCs. While there is little alternative to Windows on the client side, Linux offers an option on servers that does not depend on Microsoft’s often shifting OS road maps. Even Microsoft’s closest partner in Windows 2000, Compaq, has not hesitated to push aggressively into the Linux market.

Intel. Intel has supported Linux in two ways. First, it has invested in a number of Linux startups, including Red Hat, SuSE, VA Linux, TurboLinux, eSoft, and Cobalt Networks.[7] It also has supported Linux developers in porting Linux to Intel's next generation IA-64 microprocessor, and has provided information about the new chip to developers of Linux applications. It also has set up Itanium servers at VA Linux that are available over the Internet for Linux developers to work on (Shankland 2000).

Intel's support for Linux and its investment in Linux companies is important in establishing the legitimacy of the new OS. One likely reason is Intel's strong push its processors into the fast-growing Internet server market. Linux gives Intel a robust operating system that runs on Intel hardware and can compete against Unix for some applications where Windows NT/2000/XP is still considered less reliable. Intel also may be trying to distance itself from Microsoft, as it has never felt quite comfortable with Microsoft's first-among-equals position in the Wintel partnership — particularly after 1997, when Microsoft introduced Windows CE, a handheld OS that supports Intel and non-Intel chips alike.

Beyond the PC. IBM moved Linux beyond the PC platform when it announced support for Linux on all of its server platforms, including RS/6000, AS/400 and S/390. It has designed its implementation of Linux on the S/390 so that users can run Linux alongside the standard OS/390 — allowing IBM to compete with both Sun and Microsoft in the Internet server market, and also allow IBM's installed base of S/390 users to extend the functionality of their hardware into both Internet and business applications.

In December 2000, IBM culminated a year of strategic Linux announcements with the promise to spend $1 billion the following year on developing and promoting Linux solutions. CEO Louis Gerstner predicted the demise of proprietary solutions for mainframes and servers (Wilcox 2000). At the same time, the move to develop Linux expertise fit with IBM's ongoing shift from product sales to services, since IBM could sell consulting services whether the OS software was free or proprietary. IBM also helped found the Open Source Development Laboratory, an enterprise-focused Linux research group that in opened research facilities in Oregon and Japan after attracting more than $20 million from HP, IBM, Intel, NEC and 15 other hardware and software vendors.

[7] Dell, Oracle, IBM and others have likewise invested in Linux startups.

At the other end of the spectrum, during 2000 and 2001 a wide variety of products were announced using Linux in embedded systems. These included server appliances, Internet client appliances, hand-held personal digital assistants, and game systems. Some of the appliances use a new chip developed by Transmeta, the Silicon Valley company that employs Linus Torvalds. Torvalds himself led an effort to modify Linux to run on consumer devices without a hard disk (Markoff 2000).

Assessment of Adoption Patterns

- Adoption is Linux has been driven by the Internet. A large share of Linux use is for Internet applications, and many users are Internet-oriented businesses such as ISPs who take advantage of Linux's suitability for such applications. Also, Linux developers use the Internet as a tool for collaborative development. In its so-called "Halloween" memo, Microsoft acknowledged that the Internet provides a vital channel to coordinate development and create awareness of Linux. "OSS [Open Source Software] projects the size of Linux and Apache are only viable if a large enough community of highly skilled developers can be amassed to attack a problem. Consequently, there is direct correlation between the size of the project that OSS can tackle and the growth of the Internet." (Valloppillil 1998).
- Early Linux adopters had different characteristics from other IT users. They were younger, had stronger technical skills than most PC users, and had not built careers administering proprietary operating systems as most IT managers had. They brought Linux into the enterprise through the back door, just as an earlier generation had introduced PCs outside the reach of MIS departments.
- As a Unix-compatible operating system, Linux was able to leverage more than 20 years spent developing complementary assets co-specialized for Unix, including software, MIS department skills, university training and consulting services. At the same time, with adoption rates far beyond any previous Unix variant, Linux succeeded in providing a standardized set of APIs and features in a way that the previous vendor-specific enhanced Unix implementations had not.
- For those users with the requisite technical skills, open source reduced their dependency upon vendors and thus the vendor's negotiating power. Bug fixes did not have to wait for a vendor update, but could be developed in-house or obtained from other users. Open source also allowed

customization for local requirements, without additional licensing or support from the vendor (Leibovitch 2001).

- Companies offering complementary assets for Linux are motivated by a variety of economic and strategic incentives. The “anybody but Microsoft” sentiment is strong among companies that had lost earlier market battles with Microsoft and those who do not want to see Microsoft to extend its dominance into new markets. Others simply see a new market opportunity or a chance to leverage the development efforts of the Linux community at little cost to themselves.

VI. Conclusions

Successful promotion of a standard must balance the demands of adoption and appropriability. Achieving the widest adoption of a standard attracts suppliers of complementary assets such as software and services, which in turn fuels further adoption (Shapiro and Varian 1999). This can be achieved by widespread technology licensing on favorable terms, but by doing so, the sponsor runs the risk of losing the ability to appropriate economic rents from the standard.

In the systems era, IBM maintained tight control over all key mainframe technologies and used its own sales and marketing efforts to achieve widespread adoption among large organizations that could afford such systems. In the PC era, Microsoft and Intel each controlled a key technology standard in an otherwise open architecture, and worked with system and software vendors to achieve the widest possible adoption in a much larger market. The network era has favored even more open standards due to the need for device compatibility with underlying open network standards. IBM’s strategy has turned 180°, as the most prominent of numerous hardware makers investing in open source software to fuel product sales.

Promoters of open systems have tried various strategies to balance adoption and appropriability. One strategy is for a corporate sponsor such as Sun to make its technology widely available while relying on its ability to earn temporary monopoly rents on new innovations (Garud and Kumaraswamy 1993). Another is for a consortium of companies to establish an industry association to develop and promote a technology, as X/Open promoted a common Unix standard (Gabel 1987). The risk here is that free-riding on the part of member companies can lead to underinvestment in the core technologies of the standard.

“Open Source” software such as Linux takes another approach to the promotion of so-called “open systems.” Linux has encouraged the widest possible adoption through a liberal licensing regime which encourages copying and distribution. On the other hand, it offers a novel solution to the problem of sustaining investment in a technology standard in which there is no appropriability of new innovations. A core group of developers led by a highly visible individual (Torvalds) plays the role of sponsor, but does not represent any company or group of companies with a commercial interest in the promotion of Linux. This group instead coordinates a network of thousands of Linux developers who contribute time and effort to upgrading Linux. As analyzed elsewhere (Raymond 1999; Lerner and Tirole 2000; Hars and Ou 2001), these developers are believed to be motivated by some combination of gratification of working on challenging programming problems, career advancement and other factors. Another possible explanation has been user-driven innovation, or co-invention (von Hippel 1988; Bresnahan and Greenstein 1999). Linux programmers have improved Linux in part to solve their own problems as users, and are willing to contribute their innovations to the community in return for having access to the contributions of others.

As for the development of complementary assets, the incentives of hardware and software companies are simpler. Those companies do not profit by appropriating rents from the core technology; rather they build profitable businesses that leverage the standard. Companies such as Dell, Compaq and Gateway have done so in the Windows market and can as easily do so in the Linux market if it gains sufficient size.

Linux’s future prospects depend on the ability to maintain a single common standard while ensuring continued investment in that standard. One threat is “forking,” which could happen if companies try to appropriate economic rents by creating incompatible proprietary implementations, or individual research groups focus on specialized needs without regards to the common standard. As noted earlier, such was the norm with Unix, and this risk is inherently greater with fully open source software — although this same openness also attracts users who must customize software to fully meet their own needs (Leibovitch 2001)

At the other end of the spectrum is the risk of underinvestment due to the lack of incentives to invest in a non-appropriable standard. The continued volunteer efforts of the Linux community are vital, and could be threatened if volunteers become disillusioned by having others make large amounts of money off of their efforts. After the early success of Linux IPOs in 1999, Linux websites such as Slashdot.com hosted a heated debate over the contradictions between open source and commercialization.

There are also external threats from proprietary hardware vendors whose market share is jeopardized by Linux. As Gabel (1987) noted, even vendors who've nominally endorsed a shared standard have difficulty making credible commitments to a standard that eliminates their options for proprietary advantage. Notably, Linux threatens the business models of Sun and other Unix vendors, because low switching costs make Linux and other Unix-compatible OSs particularly attractive to existing Unix users.

But the most threatened by Linux and the open source model — and in turn its greatest obstacle — is the omnipresent role of Microsoft in the software industry. In 2001, Microsoft ratcheted up the rhetorical attacks on open source license terms by increasingly senior executives. In February, vice president Jim Allchin called such terms "an intellectual property destroyer" (Bloomberg 2001). In a May speech, senior vice president Craig Mundie referred to the GNU Public License as a "viral…threat to the intellectual property of any organization making use of it"; he also linked such software to failed ".com business models" (Mundie 2001). Three weeks later, in newspaper interview CEO Steve Ballmer said "Linux is a cancer that attaches itself in an intellectual property sense to everything it touches" (Newbart 2001). In all of its attacks, Microsoft sought to tar all open source efforts with the most restrictive terms of the GPL, rather than the less onerous terms of the BSD license used by Apple and others in open source projects. This came despite Microsoft's own incorporation of FreeBSD networking software in its Windows products — which was perfectly legal under the BSD-style licenses (Gomes 2001).

Despite these risks, Linux enjoys high growth and significant market share, which attract a healthy supply of complementary assets and gives more confidence to users considering adoption of Linux. Companies making money from Linux can invest in the continued development of the core technology, supplementing or subsidizing the efforts of volunteers. As for the Microsoft challenge, Linux may be benefiting from Microsoft's legal problems, which appears to have emboldened Microsoft's OEM customers to support alternative operating systems such as Linux.

It will not be known for a few years whether Linux or other open source projects have truly created a viable new model for standards competition. While the threats to the model should not be underestimated, we feel that the timing is favorable. The shift from personal computing to network-oriented computing is as significant a shift as the PC revolution of the 1980s, and there are opportunities for new standards and forms of industrial organization to succeed. Linux may not be the winner, but its early success has given it a first mover advantage in emerging Internet markets. If Linux does succeed, it will suggest that the open source model offers a viable solution to the dilemma of balancing adoption and appropriability, and may be more generally applicable to other network era standards.

References

Ambrosio, Johanna, "POSIX is propelled by AT & T's Unix restrictions," *Government Computer News*, March 13, 1987, 65.

Bailey, Michelle, Vernon Turner, Jean Bozman and Janet Waxman, "Linux Servers: What's the Hype, and What's the Reality." IDC Bulletin #21610, March 2000.

Bloomberg News, "Microsoft exec calls Linux a threat to innovation," Feb. 15, 2001, http://news.cnet.com/news/0-1003-200-4833927.html

Bradner, Scott, "The Internet Engineering Task Force," in Chris DiBona, Sam Ockman and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif.: O'Reilly, 1999, pp. 47-52.

Bresnahan, Timothy F. and Shane Greenstein, "Technological competition and the structure of the computer industry," *Journal of Industrial Economics* 47, 1 (Mar 1999):1-40.

Bulkeley, William M., "Linux, maverick of computing, gets respectable," *Wall Street Journal,* April 9, 2001, p. B1.

Chposky, James and Ted Leonsis, *Blue Magic: The People, Power and Politics Behind the IBM Personal Computer,* New York: Facts on File, 1988.

Egyedi, Tineke M., "Why Java™ was -not- standardized twice," *Computer Standards & Interfaces*, 23, 4, (Sept 2001), 253-265.

Free Software Foundation, "What is Free Software?", May 2000, http://www.gnu.org/philosophy/free-sw.html

Gabel, H. Landis, "Open Standards in Computers: The Case of X/OPEN." In H. Landis Gabel, ed., *Product Standardization and Competitive Strategy,* Amsterdam: North-Holland, 1987.

Garud, Raghu and Arun Kumaraswamy, "Changing competitive dynamics in network industries: An exploration of Sun Microsystems' open systems strategy," *Strategic Management Journal*, 14, 5 (July 1993), 351-369.

Gillen, Al and Dan Kusnetzky, "Linux Overview: Understanding the Linux Market Model." IDC Bulletin, International Data Corporation, 2000.

Gomes, Lee, "Microsoft Uses Free Code," *Wall Street Journal*, June 18, 2001, p. B6.

Greenstein, Shane M., "Lock-in and the Costs of Switching Mainframe Computer Vendors: What Do Buyers See?" *Industrial and Corporate Change,* 6, 2 (1997): 247-274.

Grindley, Peter, *Standards, strategy, and policy: cases and stories,* Oxford: Oxford University Press, 1995.

Grove, Andrew S., *Only the Paranoid Survive: How to Exploit the Crisis Points that Challenge Every Company and Career,* Doubleday, New York, 1996.

Hars, Alexander and Shaosong Ou, "Working for Free? – Motivations of Participating in Open Source Projects," Proceedings of the 34th Hawai'i International Conference on System Sciences, Maui, Hawaii, January 4-6, 2001.

IBM, "IBM Unveils Linux Software and Services for S/390 Server, Introducing Big Iron to Next Generation e-businesses," press release, Somers, NY, May 17, 2000.

Kawamoto, Dawn and Stephen Shankland, "VA Linux storms Wall Street with 698 percent gain," CNET News.com, December 9, 1999, http://news.cnet.com/news/0-1003-200-1489252.html

Klaus, Todd C., "Checking out Linux: looking for a good way to learn about operating system?" *UNIX World* 10, 3 (March, 1993): 66.

Kraemer, Kenneth L., Jason Dedrick and Sandra Yamashiro, "Dell Computer: Refining and Extending the Business Model with IT," *The Information Society*, 16 (2000):5-21.

Lerner, Josh and Jean Tirole, "The Simple Economics of Open Source," Harvard Business School Working Paper #00-059, February 25, 2000, http://www.people.hbs.edu/jlerner/simple.pdf.

Leibovitch, Evan, "Why Linux is like pizza," ZDNet News, March 21, 2001, http://www.zdnet.com/zdnn/stories/comment/0,5859,2695214,00.html

Liebowitz, Stan J. and Stephen E. Margolis, *Winners, Losers & Microsoft: Competition and Antitrust in High Technology.* Oakland, Calif.: Independent Institute, 1999.

Lohr, Steve, "Code Name: Mainstream; Can 'Open Source' Bridge the Software Gap?," *New York Times,* Aug. 28, 2000, p. C1

MacCormack, Alan and Kerry Herman, "Red Hat and the Linux Revolution," Harvard Business School case #9-600-009. Harvard Business School, 1999.

Malkin, Gary, "The Tao of IETF: A Guide for New Attendees of the Internet Engineering Task Force," RFC 1539, Information Sciences Institute, Los Angeles, August 1993.

Markoff, John, "Gateway and AOL Bypass Industry Stalwarts on Components," *New York Times,* May 30, 2000, http://www.nytimes.com/library/tech/00/05/biztech/articles/30chip.html

Morris, Charles R. and Charles H. Ferguson, "How Architecture Wins Technology Wars," *Harvard Business Review,* 71, 2 (March/April 1993), 86-96.

Moschella, David C., *Waves of power: dynamics of global technology leadership, 1964-2010,* New York: AMACOM, 1997.

Mundie, Craig, “The Commercial Software Model,” speech, New York University Stern School of Business, May 3, 2001.

New York Times, “Dell Bolsters Support of Windows Rival Linux,” April 7, 1999, http://www.nytimes.com/library/tech/99/04/biztech/articles/07linux.html

Newbart, Dave, “Microsoft CEO take launch break with the Sun-Times,” Chicago Sun-Times, June 1, 2001, p. 57.

OpenSource.org, “History of the Open Source Initiative,” http://www.opensource.org/history.html, 1999.

Raymond, Eric S., *The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary.* Cambridge, Mass.: O’Reilly, 1999.

Ritchie, Dennis M.; and Ken Thompson, “The UNIX time-sharing system,” *Communications of the ACM,* 17, 7 (July 1974), 365-75.

Salus, Peter H., *A quarter century of UNIX.* Reading, Mass.: Addison-Wesley, 1994.

Schneider, Wolfram, “The UNIX system family tree / BSD history chart”, v1.24, FreeBSD, Inc., April 30, 2000, ftp://ftp.freebsd.org/pub/FreeBSD/FreeBSD-current/src/share/misc/bsd-family-tree

Shankland, Stephen, **“**Linux programmers get a taste of the future,**”** CNET News.com, May 25, 2000.

Shankland, Stephen, “Linux making gains in server market,” ZDNet News, Feb. 28, 2001, http://www.zdnet.com/zdnn/stories/news/0,4586,2691286,00.html

Shapiro, Carl and Hal R. Varian, *Information rules: a strategic guide to the network economy*. Boston, Mass.: Harvard Business School Press, 1999.

SHARE, “About SHARE,” 2001, http://www.share.org/community/about.jsp

Sobel, Robert, *I.B.M., colossus in transition.* New York: Times Books, 1981.

Stallman, Richard, “The GNU Operating System and the Free Software Movement,” in Chris DiBona, Sam Ockman and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution.* Sebastopol, Calif.: O’Reilly, 1999, pp. 53-70.

Taft, Darryl K., “NBS workshop aims to ensure Posix portability,” *Government Computer News*, Nov. 20, 1987, 71.

Tanenbaum, Andrew S., *Operating systems: design and implementation,* Englewood Cliffs, N.J. Prentice-Hall, 1987.

Teece, David, “Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy,” *Research Policy* 15, 6 (Dec. 1986), 285-305.

Valloppillil, Vinod, “Open Source Software: A (New?) Development Methodology,” internal memo, Microsoft Corp., Aug. 11, 1998, http://www.opensource.org/halloween/halloween1.html

von Hippel, Eric, *The sources of innovation,* New York: Oxford University Press, 1988.

West, Joel and Dedrick, Jason, “Innovation and Control in Standards Architectures: The Rise and Fall of Japan’s PC-98,” *Information Systems Research*, 11, 2 (June 2000), 197-216.

Wilcox, Joe, "IBM to spend $1 billion on Linux in 2001," CNET News.com, Dec. 12, 2000, http://news.cnet.com/news/0-1003-200-4111945.html

Tables

Date	Unix, Open Systems	Open Source	Internet
1969	Work begins on Unix at AT&T Bell Telephone Laboratories		
1973	U.C. Berkeley licenses Unix		
1974	*CACM* publishes 1st Unix article		
1977	First BSD distribution		
1979	Multi-plaform Unix V7 ships		
1980	DARPA-funded 4.0 BSD released		U.S. Defense Dept. adopts TCP/IP standard
1981	IEEE starts POSIX committee		
1982	First Sun workstation		4.1c BSD released with sendmail
1983	AT&T defines System V interface standard		ARPANET converts to TCP/IP 4.2 BSD supports TCP/IP
1984	AT&T breakup	Project GNU begun	
1986	4.3 BSD released		IETF founded
1987	Sun/AT&T alliance Minix published with textbook		
1989	Unix factions align behind X/Open specification		HTML and HTTP created
1990	AT&T creates Unix Systems Laboratories subsidiary		ARPANET decomissioned
1991		Linus Torvalds begins work on Linux	
1992	Novell buys Unix Systems Laboratories Sun Solaris 2.0 merges BSD and ATT Unix flavors	First Slackware and SuSE Linux distributions	Internet Society formed
1993	Novell buys Unix Systems Laboratories 4.4 BSD marks end of Berkeley development	NetBSD, FreeBSD projects begun	NCSA Mosaic is first browser for personal computers
1994	Red Hat founded	BSD 4.4-Lite released First Red Hat Linux distribution	Netscape founded, releases first web browser
1995		Apache Group begins developing web server	Microsoft bundles web browser with Windows 95
1996		Linux users top 1 million	
1997		Debian develops free software guidelines	Apple bundles Microsoft browser with Mac OS
1998	Intel, Netscape invest in Red Hat	"Open Source" term coined Microsoft "Halloween" memo	Netscape announces it will release browser source code U.S. delegates Internet management to ICANN
1999	IPOs of Red Hat, VA Linux and Andover.net raise $1.1 billion	IBM, Dell announce support for Linux	America Online buys Netscape
2000	VA Linux acquires Andover	IBM pledges $1 billion for open source projects	Few buy Internet appliances in Christmas shopping season
2001	Caldera acquires SCO Unix Apple ships BSD-based OS X	Microsoft execs publicly attack open source licensing	Microsoft, Sun settle Java lawsuit

Table 1: Key events in Linux and Open Source history