How Open is Open Enough?
Melding Proprietary and Open Source Platform Strategies

Abstract:

Computer platforms provide an integrated architecture of hardware and software standards as a basis for developing complementary assets. The most successful platforms were owned by proprietary sponsors that controlled platform evolution and appropriated associated rewards. Responding to the Internet and open source systems, three traditional vendors of proprietary platforms experimented with hybrid strategies which attempted to combine the advantages of open source software while retaining control and differentiation. Such hybrid standards strategies reflect the competing imperatives for adoption and appropriability, and suggest the conditions under which such strategies may be preferable to either the purely open or purely proprietary alternatives.
The Author: Joel West

- **Professor** at KGI — the Keck Graduate Institute of Applied Life Sciences (Claremont, Calif.)
- Blogger on technology strategy, innovation and entrepreneurship
- **Researcher** on innovation and entrepreneurship
- Teacher of strategy, technological innovation and entrepreneurship
  - **Professor Emeritus** at San José State University, Department of Organization and Management
  - Lecturer at UC Irvine, Pepperdine and Temple University Japan
  - Author of teaching cases on Qualcomm, Live365 and Solar Energy
The Author (Cont.)

- **Strategy consultant** to IT companies in areas such as *open source* strategy and business model development

- Former president and co-founder of [Palomar Software, Inc.](#), a Macintosh software company

- San Diego native
  - Graduate of [Point Loma High](#), [MIT](#) and [UC Irvine](#)
The Article

- Published in 2003
- Published in Research Policy: Policy, management and economic studies of science, technology and innovation
  - http://www.journals.elsevier.com/research-policy/
- Elsevier
  - Elsevier is a world-leading provider of information solutions that help you make better decisions, deliver better care, and sometimes make groundbreaking discoveries in science, health, and technology.
  - https://www.elsevier.com/
A Tale of Two Strategies

● Tension between two standards of creation:
  ○ Appropriability
    ■ the environmental factors that govern an innovator's ability to capture profits generated by an innovation
  ○ Adoption
    ■ market share, rate of adoption
  ○ Three Studies:
    ■ Apple Computer
    ■ IBM
    ■ Sun Microsystems
Open Source: A History

- **Dynamics of proprietary platform competition**
  - Platform - consists of an architecture of related standards, controlled by one or more sponsoring firms
    - For computers includes: processor, operating system (OS), and associated peripherals
    - Some include middleware - Java, database
  - Example of what Teece (1986), links with the ability of firms to profit from their technological innovations to the appropriability regime for intellectual property rights (IPR)—either through formal de jure protection (e.g. patents) or through de facto protection such as tacit knowledge or trade secrets
    - Without IPR - firms selling a given technology can be expected to adopt marginal cost pricing and drive profit margins to zero (Katz and Shapiro, 1986; Beggs and Klemperer, 1992)
    - Without appropriability - Teece (1986) suggests that firms must use some combination of speed, timing and luck if they hope to appropriate returns generated by their innovation.
● Complementary Assets (Teece)
  ○ additional investment is required to co-specialize the asset to be useful with a given innovation, the successful adoption of the innovation and the related assets are mutually reinforcing, providing a positive feedback cycle.
  ○ For “whole solution” must attract complementary assets

● Network Externalities/ Demand Side Economics of Scale
  ○ The positive feedback, self-reinforcing cycle of success between a de facto standard and its co-specialized asset success (Katz and Shapiro).
  ○ When such network effects are coupled with switching costs between standards and high up front R&D costs, Arthur (1996) predicts that the dominant technology will enjoy “increasing returns to scale” that magnify an early lead in a technology contest.
  ○ Tipping the Contest - instability and self-reinforcing nature of such a lead; in actuality rare
  ○ Move from custom software to prepackage APIs with advent of mass-market computers
    ■ Co-specialized to work with specific platform
● **Economies of Scale**
  ○ is the cost advantage that arises with increased output of a product
  ○ arise because of the inverse relationship between the quantity produced and per-unit fixed costs; i.e. the greater the quantity of a good produced, the lower the per-unit fixed cost because these costs are shared over a larger number of goods.
  ○ may also reduce variable costs per unit because of operational efficiencies and synergies
  ○ In this case, mass-market platforms displaced more specialized products, either by providing lower cost or addressing a broader range of buyer needs (Morris and Ferguson, 1993).

● **Step 1: Product, Step 2: ??, Step 3: Profit**
  ○ Repeated evidence that operational execution is crucial to the relative success or failure of individual platforms (Morris and Ferguson, 1993; Liebowitz and Margolis, 1999; West, 2003).
  ○ Measures of platform success have focused on adoption or market share.
  ○ The more managerially relevant metric would be the sponsor’s net profit
<table>
<thead>
<tr>
<th>Category</th>
<th>Firm</th>
<th>Platform</th>
<th>Released</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mainframe</strong></td>
<td>IBM</td>
<td>IBM S/360 (S/370, S/390)</td>
<td>1964</td>
</tr>
<tr>
<td></td>
<td>DEC</td>
<td>VAX/VMS (OpenVMS)</td>
<td>1977</td>
</tr>
<tr>
<td></td>
<td>IBM</td>
<td>AS/400</td>
<td>1988</td>
</tr>
<tr>
<td></td>
<td>AT&amp;T, later OSF</td>
<td>Unix</td>
<td>1980^b</td>
</tr>
<tr>
<td><strong>Minicomputer</strong></td>
<td>Apollo</td>
<td>Domain^c</td>
<td>1980</td>
</tr>
<tr>
<td></td>
<td>Sun</td>
<td>Sun OS (Solaris)</td>
<td>1982</td>
</tr>
<tr>
<td><strong>Workstation</strong></td>
<td>Apple</td>
<td>Apple II^c</td>
<td>1977</td>
</tr>
<tr>
<td></td>
<td>Digital Research^a</td>
<td>CP/M^c</td>
<td>1976^b</td>
</tr>
<tr>
<td><strong>8-bit PC</strong></td>
<td>IBM</td>
<td>IBM PC</td>
<td>1981</td>
</tr>
<tr>
<td></td>
<td>Microsoft^a</td>
<td>Windows</td>
<td>1990^b</td>
</tr>
<tr>
<td></td>
<td>NEC</td>
<td>PC-98^c</td>
<td>1983</td>
</tr>
<tr>
<td></td>
<td>Apple</td>
<td>Macintosh</td>
<td>1984</td>
</tr>
<tr>
<td><strong>≥16-bit PC</strong></td>
<td>Palm</td>
<td>Pilot</td>
<td>1996</td>
</tr>
<tr>
<td><strong>Personal digital assistant</strong></td>
<td>Microsoft^a</td>
<td>Windows CE</td>
<td>1996</td>
</tr>
</tbody>
</table>

^a OS vendor; otherwise, vendors are vertically integrated manufacturers.

^b Widespread commercial release.

^c Discontinued.
● Mainframes: vertically integrated proprietary platforms
  ○ IBM
  ○ Others had trouble gaining market traction:
    ■ switching costs between the proprietary platforms (Greenstein, 1997)

● Personal computer brings horizontal platform control
  ○ Unexpected legal defeats on ROM copyrights, IBM lost control of its platform
  ○ Clones of IBM
  ○ IBM spent billions of dollars on proprietary technologies in an unsuccessful attempt to re-assert its leadership of the PC industry

● The King is dead, Long Live the king - The Rise of Microsoft and “Wintel”
By licensing their operating systems to multiple hardware vendors, each made their platform ubiquitous by reducing switching costs and differentiation between hardware vendors. Both operating systems shared APIs across multiple hardware vendors. As with even the most proprietary computing platform, UNIX and MS-DOS were “open” to third-party software suppliers, utilizing APIs widely disseminated to maximize software availability.

The lack of such factors made it unlikely that any manufacturer would enjoy market dominance. In Unix workstations, most of the market was fragmented among four major firms—Sun, IBM, HP and DEC (Compaq).
The Rise of Open Source

- Rather than using formal IPR protection to set boundaries between vendors and their competitors and customers, open source enlists all as collaborators, maximizing adoption throughout the value chain but minimizing the options for appropriating rents from the software

- GPL

- The War Against Microsoft

- Linux

- The Problem with Being Open
  
  ○ The problem for IBM, Apple and Sun was that by making source code freely available and modifiable, open source inherently reduced barriers to entry by rivals and switching costs by customers. So despite the appealing logic of mutual adversaries (“the enemy of my enemy is my friend”), a pure open source strategy would eliminate each company’s historic source of differentiation, their proprietary software. Each of the firms faced a dilemma of how to adapt an open source strategy suitable for their respective core competencies and resources.
<table>
<thead>
<tr>
<th>Date</th>
<th>Industry</th>
<th>Apple</th>
<th>IBM</th>
<th>Sun</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 1995</td>
<td></td>
<td>Starts working on MkLinux for Power Macintosh</td>
<td></td>
<td>Pre-release Java posted for free Internet download</td>
</tr>
<tr>
<td>October 1995</td>
<td></td>
<td>Releasing MkLinux, gives away 20,000+ CDs to ISVs</td>
<td></td>
<td>Licenses Java to Microsoft</td>
</tr>
<tr>
<td>March 1996</td>
<td></td>
<td>Buys NeXT to adapt Mach/BSD-based OS</td>
<td></td>
<td>Sues Microsoft over &quot;polluted&quot; Java</td>
</tr>
<tr>
<td>May 1996</td>
<td></td>
<td></td>
<td></td>
<td>Announces &quot;100% Pure Java&quot; initiative</td>
</tr>
<tr>
<td>October 1996</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>December 1996</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>June 1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>July 1998</td>
<td></td>
<td>Tovalds on cover of <em>Forbes</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>October 1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>August 1998</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>March 1999</td>
<td>IBM, Compaq, Oracle and Novell invest in Red Hat</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 1999</td>
<td></td>
<td>Releasing Darwin open source OS and Apple Public Source License</td>
<td>Supports Linux for workstations and PC servers</td>
<td></td>
</tr>
<tr>
<td>July 1999</td>
<td></td>
<td>Releasing Darwin source code on Apple web server</td>
<td>Certifies 4 Linux versions for PC servers</td>
<td></td>
</tr>
<tr>
<td>August 1999</td>
<td>After IPO, Red Hat reaches US$ 4 billion market cap</td>
<td></td>
<td>Releasing DB2 for Linux</td>
<td></td>
</tr>
<tr>
<td>September 1999</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>October 1999</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>December 1999</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>February 2000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>February 2000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Date</td>
<td>Industry</td>
<td>Apple</td>
<td>IBM</td>
<td>Sun</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------</td>
<td>-------------------------------------</td>
<td>-------------------------------------------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>August 2000</td>
<td>Open Source Development Lab funded</td>
<td>Releases public beta of Darwin-based Mac OS X</td>
<td>Releases tools under IBM Public License</td>
<td>Replaces own GUI with open source GNOME</td>
</tr>
<tr>
<td>September 2000</td>
<td></td>
<td></td>
<td>Purchases Cobalt, maker of Linux server appliances</td>
<td>Posts StarOffice source under Lesser GPL license</td>
</tr>
<tr>
<td>October 2000</td>
<td></td>
<td></td>
<td></td>
<td>Red Hat drops Linux support for Sun systems</td>
</tr>
<tr>
<td>November 2000</td>
<td></td>
<td></td>
<td></td>
<td>Releases Solaris source under SCSL license</td>
</tr>
<tr>
<td>December 2000</td>
<td></td>
<td></td>
<td>Announces IBM will spend US$ 1 billion on Linux in 2001</td>
<td></td>
</tr>
<tr>
<td>January 2001</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>March 2001</td>
<td>Microsoft launches “Shared Source” initiative</td>
<td>Ships Mac OS X, based on Darwin and FreeBSD</td>
<td>Launches “Peace, Love and Linux” ad campaign</td>
<td></td>
</tr>
<tr>
<td>June 2001</td>
<td>Microsoft CEO Ballmer says “Linux is a cancer”</td>
<td>Hires FreeBSD co-founder</td>
<td></td>
<td></td>
</tr>
<tr>
<td>November 2001</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>April 2002</td>
<td>HP wins Linux-based supercomputer contract with US Department of Energy</td>
<td>With non-profit Internet Software Consortium, co-sponsors OpenDarwin.org</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 2002</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>September 2002</td>
<td></td>
<td></td>
<td>Red Hat Linux expands support to all IBM servers</td>
<td>Introduces LX series Linux-based PC servers</td>
</tr>
</tbody>
</table>
Apple: Re-Use and Leverage

- Failed at its own OS
- Market share built on proprietary GUI, grew until September 1995
  - Windows 95
- In 2 years - US$ 2 billion in losses and forced resignation of two Apple CEOs
- Jobs back
  - Purchases NeXT - NeXTStep OS
  - Unix variant that combined the Mach operating system “kernel” with other components from BSD Unix
    - added extensions, GUI, software development and sys admin tools
  - MkLinux - Mach Linux kernel combination
  - Mac OS X
  - Darwin - Mach, FreeBSD and some NeXT components as a new open source operating system
Apple: Re-Use and Leverage

- Ernest Prabhakar - Project Manager:
  - We realized that the pieces they’re most interested in are the most commoditized. There wasn’t any proprietary technology added that we had to worry about them copying. . . . We started making the case [that] we should just open the source code and release it as a complete BSD-style operating system. (Wayner, 2000, p. 175)

- “embrace and layer” by building Apple’s proprietary code on top of the publicly shared open source code

- BSD-style License

- Most enduring controversy continued over Apple’s decision to hold some layers of its operating system entirely proprietary. Most of the public Darwin source was a derivative work of the public FreeBSD
  - Withheld 75% - GUI, NeXT and Mac OS application support, Truetype, Quicktime

- By 2002, 48 contributors
Apple: Re-Use and Leverage

● **Pros:**
  ○ Leverage off the larger BSD communities to incorporate enhancements in networking and other technologies, and to port Unix-based applications such as web and mail servers
  ○ Low-level documentation to third-party hardware vendors, freeing Apple to concentrate its support efforts on application software
  ○ Retained differentiation in the traditional areas where it had mattered most—in graphics and ease of use for its core markets in graphical design and education

● **Cons:**
  ○ Less valuable to user-contributors
  ○ Fewer users that contributed to the Darwin sources, the less benefit
IBM: From Platforms to Applications

- On investing US$1 Billion in Linux, CEO Louis Gerstner:
  - The movement to standards-based computing is so inexorable, I believe Sun—and EMC and Microsoft for that matter—is running the last big proprietary play we’ll see in this industry for a good long while. (Wilcox, 2000)

- Anything’s better than a world run by Microsoft

- Shift in leadership traumatic

- Billions of dollars on OS/2 and joint ventures with various industry rivals to reassert dominance and/or break free from Microsoft

- Lost “OS Wars”
  - Focus on end-to-end delivery with client server/ and software solutions
IBM: From Platforms to Applications

- First Open Source Initiative:
  - Integrate corporate mainframes (with their vast legacy databases) to directly support e-commerce and intranet initiatives
  - WebSphere product family
    - built on Apache open source web page server
    - Resulted in subsequent collaboration and funding of Apache
      - including adaptation for use on Windows NT
    - IBM engineers contributed code back for use by all Apache implementations
    - Working with (largely user driven) open source group provided more flexibility than using someone else’s proprietary solution
      - WebSphere first IBM application available on multiple Linux platforms
        - small cost to provide Linux version
IBM: From Platforms to Applications

- WebSphere product indirectly led to IBM’s November 2001 formation of Eclipse
  - Touted as a universal tool platform
  - CPL - allowed commercial distribution of derivative works
  - IBM merged updates from it into its commercial product, WebSphere Studio Workbench

- Development of Applications follows 3 common threads:
  - IBM accepted commodization of certain layers of its application architecture and was thus willing to collaborate with open source software programmers to make a shared technology available to all; these layers typically implemented open Internet standards which offered less opportunities for differentiation
  - In many cases the shared software competed with proprietary solutions developed by Microsoft using its US$ 4 billion annual R&D budget, such as its Internet Information Services web server
IBM: From Platforms to Applications

- The shared software was released under a non-GPL license allowing IBM to retain technology or make proprietary enhancements

- IBM and Linux - A Long Time Coming
  - Through the 90’s, no intention of supporting Linux
  - In 2000, announced support for entire product line
  - Advantages
    - Meet customer demand
    - Linux provided a common set of APIs across its entire product line, providing a unified architecture for software developers
    - The comparatively immature (yet complex) operating system required support services, a traditional IBM strength
IBM: From Platforms to Applications

- Wladawsky–Berger:
  - We’ve wedded ourselves to the integration of the solution, the notion being that the Internet and e-business solutions are more important than any particular component. And as a result, we’ve changed all our business models so that the integration of the pieces has become more important than any one piece. (Cooper, 2001)

- Linux allowed IBM to make changes to improve its hardware differentiation for enterprise customers

- Open Source a “logical extension of IBM culture”
  - released source for Andrew File System (OpenAFS) and Jikes Java compiler

- Both under IBM Public License, a predecessor to the Common Public License
Sun: Opening New Platforms (New Platforms)

- Sun OS (later Solaris), Unix-based operating system
- Faced threats from Linux and Windows NT
  - As 90’s ended fought for market share and to retain control of proprietary technology
- Historically emphasized support for “open architecture”, it used proprietary extensions to Unix software to differentiate itself from workstation rivals such as HP, IBM and DEC (later Compaq)
- To improve adoption, it licensed its workstation and OS technology to customers and complementors; this included a small number of makers of “clone” products, most notably in Japan (Garud and Kumaraswamy, 1993)
- Retained full control of the architecture, allowing it to rapidly evolve the technology rather than negotiate with standards committees
- Proprietary but open model
- Originally, Sun lacked such a “co-opetition” relationship with Microsoft, but after Windows NT found themselves in direct competition
- Windows NT and Linux both a low-cost threat to Sun:
  - Both were based on high-volume Intel processors, whose performance was increasing more rapidly than that of Sun’s proprietary RISC processors
  - Windows presented a more proprietary approach under control of a strong, centralized rival
  - Linux offered greater openness that was supported (initially) by a diffuse group of hobbyists
- Goal in the 90’s: New platforms to detract from Microsoft
Sun: Opening New Platforms

- Originally, Sun lacked such a “co-opetition” relationship with Microsoft, but after Windows NT found themselves in direct competition.

- Windows NT and Linux both a low-cost threat to Sun:
  - Both were based on high-volume Intel processors, whose performance was increasing more rapidly than that of Sun’s proprietary RISC processors.
  - Windows presented a more proprietary approach under control of a strong, centralized rival.
  - Linux offered greater openness that was supported (initially) by a diffuse group of hobbyists.

- Goal in the 90’s: New platforms to detract from Microsoft.

- Most of Sun’s efforts went towards establishing Java as a new platform with a common set of APIs available on a wide range of computer systems, under the slogan “write once, run anywhere.”
  - 4-year lawsuit accusing Microsoft of trying to hinder Java’s success, a lawsuit settled out of court in January 2001.
Sun: Opening New Platforms (Partly Open Source)

- Attempted to establish Java at ISO/IEC Joint Technical Committee 1 (JTC1) and EMCA
  - Withdrawn after vigorous and well-financed opposition from competing computer makers, particularly Microsoft and HP (Egyedi, 2001)
  - Also objected to provisions that would have required it to surrender IPR to the standardization committee

- In August 1999, spent US$ 73.5 million to acquire the German maker of StarOffice (later OpenOffice and finally LibreOffice), a clone of Microsoft Office

- Preoccupation with its offensive strategy against Microsoft contributed to its failure to defend against Linux

- Moves against Sun by Microsoft and other companies pushed them towards open source solutions improving it’s ability to compete and cooperate on other open source endeavors
Sun: Opening New Platforms

- Partly Open Source
  - First open source strategies focused on its competition with Microsoft in getting core technologies adopted by users and software developers
    - Pushed Java in StarOffice and in core of its own product, Solaris
    - Reacted towards push for openness in Java with more licensing and standardization committees
  - GPL would provide the fastest path towards adoption but co-founder Bill Joy explained:
    - I can't license all of Sun’s intellectual property under the GPL, because it just won’t work. I don’t see any reason why I should give somebody who’s doing commercial reuse unfettered access to stuff that cost me millions of dollars to do. We’re spending over a billion dollars a year in research. I can’t just throw it all on the street. . . . If I make code available under the GPL, I’ll lose control of it. . . . The GPL just doesn’t solve my business problem at Sun. I would like all of our intellectual property to be available in source form, but I can’t economically do that under the GPL. (Kim, 1999)
Sun: Opening New Platforms

- February 1999, released Java under Sun Community Source License
  - Hybrid between a traditional proprietary license and a BSD-style open source license
    - Right to modify the source code
    - Royalty free distribution in open source projects
    - Royalties for commercial redistribution
    - Testing requirement to maintain compatibility and prevent forking

- “[P]rovides protection for intellectual property, . . . guarantees structured innovation within a single responsible organization, [and provides] clear control over compatibility” (Gabriel and Joy, 1998)
  - The SCSL processes in many ways resemble a formal standards consortium or de jure standardization committee more than the “bazaar” associated with decentralized Linux development

- In December 2000, released Solaris under SCSL

- For StarOffice, released all source code to OpenOffice in October 2000
  - Licensed in a way that guaranteed the source would always remain public, but allowed its use in commercial products by Sun or anyone else

- Severely lagging behind Microsoft Office
Sun: Opening New Platforms (If You Can’t Beat ‘Em…)

- In December 2000, released Solaris under SCSL
- For StarOffice, released all source code to OpenOffice in October 2000
  - Licensed in a way that guaranteed the source would always remain public, but allowed its use in commercial products by Sun or anyone else
  - Severely lagging behind Microsoft Office
- Sun did little to embrace Linux
- Initially ambivalent
  - Solaris a proprietary version of Unix therefore a world where Linux was the norm would eliminate that advantage
- Eventually adopted Linux and purchased Cobalt
- 2 years later announced Linux and Unix compatible solutions
Effect of Open Source on Platform Strategies

- Not surprisingly, the response of leading industry firms varied depending on whether they had used software as a source of competitive advantage, and whether they retained other sources of competitive advantage.

- Apple vs IBM vs Sun
  - IBM
    - Usually sold software in combination with its hardware
    - Continued to differentiate itself based on services such as integration, services that would provide switching costs if it adopted commodity software
    - Key revenues were in mainframe and midrange systems where there were few remaining competitors
  - Linux offered IBM something it had never had—a common set of software APIs across its entire product line
  - Apple and Sun faced diminished profit margins if they shared the same software as their rivals (or vice versa)
    - Both opened source code from OS (unlike IBM)
      - Opening parts:
        - In part by building on code that was already open source, Apple chose to grant all rights to a subset of its new OS X operating system
      - Partly Open:
        - Sun released the entire source of Java and Solaris under restrictive terms—the former to improve adoption, the latter in response to competition from the open source Linux
Effect of Open Source on Platform Strategies

- Apple and Sun faced diminished profit margins if they shared the same software as their rivals (or vice versa)
  - Both opened source code from OS (unlike IBM)
  - Opening parts:
    - In part by building on code that was already open source, Apple chose to grant all rights to a subset of its new OS X operating system
  - Partly Open:
    - Sun released the entire source of Java and Solaris under restrictive terms—the former to improve adoption, the latter in response to competition from the open source Linux
Microsoft Speaks

● Most to lose by Open Source
  ○ As a result of its success, faced least pressure to open source code
  ○ Mounting pressure from users switching to Linux forced a response

● Publicly attacked the movement, particularly the “viral” nature of the GPL
  ○ Later clarified to support BSD-style licensing

● Unveiled “Shared Source” - Microsoft’s own source code license
  ○ Its strategy had evolved to allow PC vendors, third-party developers and large end-users to view but not modify the source to Windows
  ○ To win the hearts and minds of academics, Microsoft also allowed universities to both view and modify the source for internal research
Microsoft Speaks

• Clearly a catalyst for open source
  ○ criticized for using BSD networking in Windows and then making it incompatible with Unix-like systems
  ○ Bill Joy complained:
    ■ The top predator now is Microsoft. We didn’t have a top predator back when I did TCP/IP. When you have a person with unlimited funds who is clearly focused on destroying the value proposition of what you’re doing, you’d be a fool not to account for them in the strategy that you adopted. (Kim, 1999)
Discussion

- Study suggests three stage evolution:
  - Began as vertically integrated to provide complete solutions
    - Prefer proprietary strategies
      - better barriers to imitation
      - better margins
      - only available to one or two market leaders
    - Proprietary solutions can become infeasible for a number of reasons
      - market share lower than the minimum efficient scale necessary to support proprietary R&D
      - not enough market power to resist buyer demands for open standards
      - “tipping” of the standards contest in favor of the open standard, making it infeasible to establish (or maintain) a proprietary standard
      - decision to accept commodization of the particular architectural layer and shift competitive advantage to another architectural layer
    - May be cost-effective, but may prove difficult if counter to company culture
      - Apple, IBM, Sun all forced to open source to compete
Discussion

- Proprietary firms simplify their technical and business decisions because they control the environment and do not require interoperability.

- When that fails:
  - forced to work open with open standards to achieve interoperability
    - requires technical investment
    - less lock in with customers
  - Without innovation and proprietary lock-in to provide barriers to entry and imitation, invariably firms will find it difficult to achieve competitive advantage with these new strategies. Among the functional strategies that the three firms used include marketing, customer service, product design, engineering efficiency and leveraging previously established brand name reputations; the long-term viability of all these strategies have yet to be proven.

- A vendor’s decision to disclose technology is an irrevocable waiver of its ability to appropriate the returns from that technology.
  - Experimenting to find the right compromise that maximizes gains and profit
  - The two emerging strategies are:
    - Opening parts—waving control of commodity layer(s) of the platform, while retaining full control of other layers that presumably provide greater opportunities for differentiation
    - Offensive strategy to speed adoption of a new platform-related standard or a particular implementation of such a standard
    - Partly open—disclosing technology under such restrictions that it provides value to customers while making it difficult for it to be directly employed by competitors
    - Waiving intellectual property rights makes the standard (or implementation) more attractive to competitors and key users, priming the positive-feedback bandwagon effects that can accrue to early market leaders
    - Also increases the number of products that are interoperable with the vendor’s products, particularly important for networking and other communications standards.
Discussion

- A vendor’s decision to disclose technology is an irrevocable waiver of its ability to appropriate the returns from that technology
  - Experimenting to find the right compromise that maximizes gains and profit
  - The two emerging strategies are:
    - Opening parts—waving control of commodity layer(s) of the platform, while retaining full control of other layers that presumably provide greater opportunities for differentiation
      - Offensive strategy to speed adoption of a new platform-related standard or a particular implementation of such a standard
      -Waiving intellectual property rights makes the standard (or implementation) more attractive to competitors and key users, priming the positive-feedback bandwagon effects that can accrue to early market leaders
      - Also increases the number of products that are interoperable with the vendor’s products, particularly important for networking and other communications standards
Discussion

● Partly open—disclosing technology under such restrictions that it provides value to customers while making it difficult for it to be directly employed by competitors

● For both strategies:
  ○ Allow “sophisticated” users to help improve products
Fig. 2. User rights under open and quasi-open source licenses.
Future Platform Strategies

- This study focused on three approaches:
  - Vertically integrated proprietary systems, as represented by the IBM 360 (Chandler, 1997; Moschella, 1997)
  - Platforms assembled from proprietary layers that are freely licensed to all, such as “Wintel” architecture (Morris and Ferguson, 1993; Grove, 1996)
  - de jure standards not sponsored by any single firm but shared by all, epitomized by “open systems” and Europe’s GSM digital telephone standard (Gabel, 1987; Funk and Methé, 2001)

- Open source standards differ from other unsponsored open standards mainly in degree, to the extent that the entry and imitation barriers are dramatically lower

- Shared standard—with the associated implications for governance and differentiation—is not fundamentally different between the open source Linux or FreeBSD and its open systems (Unix clone) ancestors
Future Platform Strategies

● Hybrid approach existed for decades
  ○ ex. ISO character sets, ANSI C, Ethernet, TCP/IP

● Sun and Apple blur the lines between the proprietary and unsponsored standards
  ○ By retaining an element of control, they retain many of the competitive benefits of sponsorship
  ○ But, by reducing duplicative R&D they can create shared communities that in many ways are indistinguishable in practice from nominally unsponsored standards—assuming that the sponsors move aggressively enough to build sizable communities of adopter/collaborator

● Open source strategies studied also call attention to the use of platform extension as a strategy to deal with commoditization of lower-level platform layers
  ○ extension normally involves developing additional application or “middleware” as the highest level standards layers of an architecture
Future Platform Strategies

● Such vertical integration into applications suggests at least a partial re-examination of the assumption that platforms succeed through their ability to attract a supply of third-party applications

○ Gallagher and Park (2002) have shown that in-house applications development was crucial in deciding a success of platform contests in the videogame console industry

○ If applies generally, firms need to garner the financial resources to supply a complete (or at least basic) supply of complementary assets expected by adopters, rather than building early market share perceptions to attract third-party suppliers of such assets

● In other cases, the attempt to differentiate may continue not to higher architectural layers, but with system integration or design

○ Nokia/Symbian
Implications for Open Source

- Open source provides few direct benefits to the vast number of users who lack the requisite technical skills to do their own development, but instead is best suited for technically proficient users (such as Internet service providers) with strong motivations for customization (West and Dedrick, 2001)

- The degree to which open source adds value beyond this niche depends on how much it enables other attributes more directly valued by users, such as greater reliability, lower cost or expanded variety of complementary assets
  - ex. indirect benefits - the provision of complementary assets
    - Normally, the provision of applications for an operating system is controlled by the formal, published interfaces (Langlois and Robertson, 1995; West and Dedrick, 2000)
    - However, in an open source system, a third-party software supplier can add its own interfaces as needed to provide functionality unanticipated by the original author of the OS
Implications for Open Source

○ Another postulated indirect user benefit of open source systems is increased reliability through the concurrent debugging efforts of a widely distributed community of user-programmers (Raymond, 1999)
  ■ Important for all three firms studied
  ■ Consistent with users getting involved only when there is a problem of great concern to them

○ Research on large, successful open source projects such as Linux and Apache assumes away variance in what may be a key independent variable: size of the user-programmer community
  ■ Major reason open source projects fail is a lack of user-contributors to do the work
  ■ Vendors face a particularly important adoption challenge
  ■ Attract enough of the right sort of users early enough to improve the quality and features of the software
Implications for Open Source

- How applicable are these benefits to the hybrid strategies?
- For the “opening parts” strategy, must the open part of the platform have value on its own to win enough user-programmers? Or is it enough that it be part of a larger system of crucial importance?
- For partly-open, the question is, whether the strategy provides a stable enough allocation of the returns of innovation between software vendors and users?
A Roadmap to Proliferate Open Source Software Usage within SA Government Servers

- OSS is increasingly being recognized by the government sector around the world
- The South African government has acknowledged that OSS is a viable choice for proprietary software
- Started deploying OSS within departments in 2008
- Not only South Africa, but many governments (Malaysia, Germany, and UK) around the world have investigate and realize the benefits of using OSS
Key point of this paper

- Elaborates the definition, history and benefits of OSS
- Showing the usage of OSS on the server side in the SA government
- Analyze the static report on the current implementation of OSS within the SA government servers
- Conclude a roadmap that can be used to accelerate OSS usage in government network servers
Benefits of OSS

• **Low software costs**
  • The main attraction, there is a huge cost difference between OSS and proprietary software
  • Many OSS are not require License fees and upgrade fees

• **No Vendor Lock-in**
  • Users can decide change the supplier or service without worrying about changing the software currently being used
  • Governments moving towards OSS in order to ensure that their data and software can be accessed across different platform

• **Localization**
  • The availability of the source code enables developers to customize the software as desired. (Ex: Can translated to local languages)
Benefits of OSS Cont.

- Speed of deployment
  - OSS can be developed by large group of people in order to allows for OSS applications to be released and deployed much faster as compared to proprietary software

- Enhance Security
  - OSS is considered to be secure as it is developed by numerous collaborating developers who are able to detect and fix bugs rapidly.
OSS alternatives on the Servers

• Linux Operating System
  • It is the most popular open source project available to date
  • The Linux operating systems have been found to be more secure and less vulnerable to viruses as compared to Microsoft Windows

• Apache Web Server
  • Market share of above 50% of the world’s web servers.
  • Caters for both Microsoft and Linux OS and has been very successful

• Sendmail
  • It is being used on 67% of the world’s electronic mail servers suggesting that it is a standard for mail servers.
Statistic analyze for usage

- Mix operating systems are being deployed in government
- Microsoft Windows is mostly used with 29 out 31 (94%) departments indicating usage.
- Linux operating system is also highly favored with 74% of the departments indicating usage.

the types of operating systems using in SA government
Statistic analyze for usage

- Other OSS server applications are rarely used as proprietary software still dominates the server environment.
- Only MySQL and Apache are extensively used on the servers while proprietary software still dominates on most servers in government.
Statistic analyze for usage

- 58% of the departments are using proprietary software for network management
- Others are available as alternatives for proprietary software
- OSS has always been considered to be more reliable and secured especially when it comes to the network environment
Statistic analyze for usage

- The results show that OSS applications are rarely used by the departments when it comes to protection against malicious code software.
Statistic analyze for usage

- Security applications are mainly based on proprietary software.
- Although OSS offers a variety of useful benefits to the government sector and the private sector in general, it is still not fully implemented within the SA government.

Other OSS security tools used within government departments and agencies.
Roadmap for OSS proliferation
Conclusion

- The implementation of OSS by the government sector is one of the enablers to the adoption of OSS by the private sector.
- Little has happened with regards to OSS implementation in the SA government.
- OSS is partially used within different government servers.
- Various reasons such as OSS lack of compatibility and difficult migration plans have been cited as the main challenges for fully deploying OSS.
- The roadmap is intended to be utilized by government departments to grow OSS implementation on the server side in order to realize the benefits that come with using OSS.
- Roadmap hasn’t been evaluated practically