The Effect of Computerization Movements Upon Organizational Adoption of Open Source

Joel West
<Joel.West@sjsu.edu>
College of Business
San José State University
One Washington Square, San José, CA 95192-0070 USA
+1-408-924-7069; fax: +1-408-924-3555

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

To be presented at
Social Informatics Workshop:
Extending the Contributions of Professor Rob Kling
to the Analysis of Computerization Movements
March 11-12, 2005

Abstract:

The free and open source software (F/OSS) computing movements have argued that F/OSS projects lead to better software, freedom from vendor control, and social benefits by sharing software and its associated source code. While these movement grew out of the interests of programmers to write better software for their own purposes, the open source movement has focused on gaining widespread adoption of F/OSS by businesses and other organizations. This requires acceptance by IT organizations and professionals, whose views of F/OSS have been largely ignored in prior research. We interviewed twenty one IT professionals in fourteen business and public sector organizations to uncover their views on F/OSS and the extent of adoption by their organizations. We find that most users do not value access to source code, and very few have ever modified source code even when they use open source software. Users are much more interested in the low cost of F/OSS, which the movement downplays, and there is no consensus about the relative quality of open source versus closed source software. The main point of agreement is on the importance of control and choice, and the freedom from vendor lock-in, that comes with F/OSS. Finally, we find that users are generally agnostic about the ideologies of the F/OSS movement, but that in some cases movement advocates act to encourage adoption within their organizations.

Revised as of February 28, 2005

1. Introduction

The emergence and adoption of new technologies can lead to important social changes. While social change may occur as an unintended by-product of technological change, advocates of new technologies often have promoted them as instruments of positive social change. One example is computer technology, which has been championed by so-called "computerization movements" (CMs) (Kling and Iacono, 1988).

A CM is defined as "a kind of movement whose advocates focus on computer-based systems as instruments to bring about a new social order." (Kling and Iacono, 1988: 227). Members of CMs advocate the adoption of specific computing uses and arrangements such as artificial intelligence, office automation, computer-based education, personal computing, and internetworking (Kling and Iacono, 1988; Iacono and Kling, 2001).

Two computerization movements that have gained considerable attention in the past decade are the Free Software and Open Source Software movements — often referred to by the acronym F/OSS. These overlapping (but distinct) movements use a combination of mobilizing rhetorics. Some are framed in ideological terms of social transformation, allowing developers to study, improve and redistribute software freely for the good of the broader community (Stallman, 2002). The movements also employ more utilitarian rhetorics, claiming F/OSS is a superior approach to software development that produces higher quality code much faster than commercial software projects (Raymond, 2001).

These movements also have other distinguishing characteristics. Unlike other CMs, the F/OSS movement does not promote computerization *per se*, but assumes that computerization is widespread, and attempts to recruit more F/OSS developers and users (Elliott and Scacchi, 2004). These movements are also the latest stage of a broader Open Software movement that extend back to 1969 with the creation of the Unix operating system. Finally, these movements require cooperation between two distinct groups — F/OSS developers and adopters. So far, there has been considerable research on the organization and motivation on F/OSS developers (Lerner and Tirole, 2002; Markus, et al., 2000), but little on the motivations of current or potential users.

In our study, we ask the following research questions:

- What are the ideology, beliefs, motivations of F/OSS movements?
- Are these beliefs shared by potential organizational adopters of F/OSS?

- What role do members of the OSS movement have as change agents in getting IT organizations to adopt open source software?

We first briefly review the history of the F/OSS and earlier movements. We then describe the results of an empirical field study of professional IT managers and their attitudes towards F/OSS adoption. From this, we draw conclusions about relevance to adopters of the mobilizing rhetorics used by the Free Software and Open Source movements.

2. Three Decades of Open Software Movements

West & Dedrick (2001) show how during the 1990s, the technology and business strategies of the Free Software and Open Source movements grew directly out of the Unix ecosystem of the 1970s and 1980s. In particular, the Linux operating system and Project GNU emulated many of the technical advances of Unix and leveraged its associated supply of complementary assets.

At the same time, there are important differences between the two F/OSS movements, which Stallman (2002) terms "two political camps." [1] There were also similar divisions within the earlier Unix movement, particularly with regards to motivating and organizing innovation. As such, the Unix (Open Systems), Free Software and Open Source movements represent three distinct but related efforts within a larger movement we collectively label the "Open Software" movements (Table 1).

Unix: Academic Research and Open Systems

The open source and free software movements are a direct outgrowth of the development of the Unix operating system, and its associated movement. The Unix movement provided standards essential to later F/OSS developers, widely dispersed source code under F/OSS licenses and originated some of the same goals (notably reduced vendor control) that later became pillars of the F/OSS movement.

Beginning in 1969, the Unix operating system was developed by AT&T computer science researchers for use by themselves and their peers. AT&T was forbidden from being in the computer business, so the research project was licensed (in source code form) to users throughout the world. With the source available, sophisticated users made their own changes: by 1976 users had begun to circulate bug fixes and

[1] With regards to IP strategies, the term "Open Source" is a proper superset of "Free Software", but because of the differences in movement goals we use "F/OSS" to refer to these two movements. European members of the F/OSS movement often include "Libre" in the acronym ("FL/OSS"), but we have not found an example where "libre" is not synonymous with "free software" as defined by Stallman (1999: 56).

other improvements, some of which were eventually included by AT&T in subsequent releases (Salus, 1994).

Despite the restrictions on its distribution, Unix gained a loyal following for two reasons. First, in making good tools for their own software development, the AT&T researchers created a system that appealed to a wide range of computer scientists. Secondly, almost from the beginning the operating system was designed for portability between hardware systems, the only major operating system to provide such flexibility.

From this, there were two major submovements which diffused Unix to non-AT&T users during the 1980s and 1990s. First, led by U.C. Berkeley, Unix became a major basis for computer science research in the U.S. and elsewhere. Second, the portable operating system fueled the vendor-independence goals of the Open Systems movement.

BSD-related Academic Research. Of all the research and other improvements to Unix outside AT&T, the largest proportion came from the Computer Science Research Group at the University of California at Berkeley. University programmers began to develop their own supplemental code for Unix, shipping the first Berkeley System Distribution (BSD) in 1977. From 1980-1994, the Berkeley researchers shipped their own BSD enhanced Unix distribution with source code to sites with a valid AT&T Unix license (Salus, 1994, West & Dedrick, 2001).

But the requirement to hold an AT&T license limited the adoption of BSD, particularly by individuals and smaller businesses. In 1990, Berkeley staffer Keith Bostic began soliciting volunteers via the APRAnet to re-implement the AT&T code.[2] In 1994 Berkeley released 4.4BSD-Lite, the first Unix implementation not encumbered by an AT&T license. This code formed the basis of the NetBSD and FreeBSD open source projects, which were soon available under the BSD license for free download on the Internet (Morin, 1995; McKusick, 1999; West & Dedrick, 2001).

At MIT, researchers used the BSD operating system as the basis for developing a graphical user interface. The result was X11, which was released in source code form and became the basis of nearly all

[2] One of the rewritten packages that Bostic solicited was a lightweight database that became known as BerkeleyDB. In 1996, to support and eventually commercialize this software Bostic co-founded the open source company Sleepycat Software (interview, 2/4/2005)

subsequent Unix (and Linux) graphical interfaces. The project was largely funded by cash, equipment and developers donated by Digital Equipment Corporation and IBM.

Open Systems. The Open Systems movement was driven by a desire of users to reduce IT vendor power associated with control over *de facto* standards and high switching costs. It was supported by IT vendors that had been unsuccessful in establishing their own proprietary *de facto* standards. To allow buyers to easily switch between or integrate systems from competing vendors, the movement pushed for open standards, notably for application software (both internally developed and third party) and computer networking. But unlike the academic research, the openness was manifested by open standards, with software still developed and distributed as proprietary implementations (Gabel 1987; Grindley, 1995; West, 2003).

An early linchpin of the movement was abstracting the range of Unix implementations to create a formal compatibility standard to increase application portability between operating systems. Beginning in 1984, the IEEE sponsored the POSIX (Portable Operating System Interface) standardization effort (Isaak, 2005).

Other key technologies of the Open Systems movement were based directly on the earlier academic research, including the TCP/IP network code developed for BSD Unix. BSD's most visible leader, Bill Joy, left Berkeley in 1982 to co-found Sun Microsystems, the first large IT vendor to fully committed to an Open Systems strategy (Garud and Kumaraswamy 1993). Sun later played a major role in promulgating Unix adoption and shaping Unix standards. However, rival standards from competing vendor factions limited portability and reduced user adoption (Grindley, 1995).

To summarize, key characteristics of the Unix-related movements included:

- Programmers writing software for their own needs, i.e. for others like them
- User access to source code, which allowed quick error correction and customization;
- Free (price) distributions (under BSD and MIT licenses) of software developed for research projects.
- Direct vendor sponsorship (as at MIT) of public software research.
- Efforts by users to increase control of IT infrastructure and reduce IT vendor power, by reducing the vendor lock-in created through switching costs.

"Free Software"

Of the related Open Software movements, the Free Software movement is the most overtly ideological. Its founder, Richard Stallman, explained the philosophy behind the movement:

> Every decision a person makes stems from the person's values and goals. People can have many different goals and values; fame, profit, love, survival, fun, and freedom, are just some of the goals that a good person might have. When the goal is to help others as well as oneself, we call that idealism.
>
> My work on free software is motivated by an idealistic goal: spreading freedom and cooperation. I want to encourage free software to spread , replacing proprietary software that forbids cooperation, and thus make our society better. (Stallman, 2002, Ch. 15)

Today Stallman's story is widely known — of how he worked as a programmer at MIT, found it increasingly frustrating to be unable to modify (or share modifications of) proprietary software, of how he eventually quit in 1984 to start Project GNU (cf. Stallman, 1999, 2002). The next year he founded the Free Software Foundation, a non-profit corporation to support his nascent "Free Software" movement. By his own telling, Stallman's ethos was formed by his early career developing academic software:

> Whenever people from another university or company wanted to port and use a program, we gladly let them. If you saw someone using an unfamiliar and interesting program, you could always ask to see the source code, so that you could read it, change it, or cannibalize parts of it to make a new program (Stallman, 1999: 53).

From this he began his campaign for "software freedom", to have "free software" supplant all proprietary software:

> "Free software" is a matter of liberty, not price. To understand the concept, you should think of "free" as in "free speech," not as in "free beer." Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software (Free Software Foundation, 2005).

To define "free software", Stallman derived four levels of "software freedom," oriented towards the needs of fellow programmers (Table 2). As a practical matter, that meant that Stallman and his FSF allies judged software licenses and business models to indicate which ones conformed to the norms of the movement. The software freedom principles were codified in the GNU General Public License, which

imposes a doctrine of compulsory sharing (also known as “copyleft”) in exchange for free use and modification rights for the source code (Rosen, 2004; West, 2005).

The goals of Project GNU were to create an entire Free Software operating system, which from its earliest days was conceived as an unlicensed Unix clone. The project began by making line-oriented software development tools for other programmers, including the emacs text editor and the gcc compiler.

In 1991, college student Linus Torvalds also wanted a Unix that was cheap to run on his own PC. After being dissatisfied with Minix, he started to write his own, using both Minix and prior Unix standards to write something that would run existing Unix applications. He based the operating system on the GNU tools and after a year licensed Linux under the GPL (Moody, 2001: 31-50). By the time Torvalds was on the cover of *Forbes* magazine (August 1998), Linux had become the success story of the Free Software movement. The success of Project GNU and particularly Linux created visibility for GPL among sympathetic programmers. Thus, as of February 2005, 66.3% of the projects on SourceForge.net used the GPL, while the second popular license was the library version of the GPL with 10.7%.

The focus by Stallman and allies on the ideology and purity of the Free Software movement brought both attention and controversy. Like other movements, it sought to redefine language to advance its cause — as when Stallman declared war on terms such as “intellectual property” and “Linux”. But Stallman’s efforts to rebrand the Torvalds effort as “GNU/Linux” stirred animosity between the two men (as captured by Moore, 2002). More fundamentally, Torvalds explicitly rejected the Free Software ideology; in one famous exchange, he told a critic “Quite frankly, I don't *want* people using Linux for ideological reasons. I think ideology sucks.” (KernelTrap.org, 2002). Thus, in 1998 key Linux supporters began a new movement, one driven by practical rather than ideological goals.

“Open Source” Software

In January 1998, Netscape announced it would release on the Internet the source code to its web browser. In response, in February a group of Linux advocates met in Silicon Valley to strategize ways to promote the adoption of Netscape, Linux and other similar technologies by mainstream businesses (Raymond 1999). They saw the mobilizing rhetoric of the “free software” movement as a major obstacle:

> [W]e have a problem with the term "free software", itself, not the concept…. The problem with it is twofold. First, it's confusing; the term "free" is very ambiguous (something the Free Software Foundation's propaganda has to wrestle with constantly). Does "free" mean "no money charged?" or does it mean "free to be modified by anyone", or something else?
>
> Second, the term makes a lot of corporate types nervous. While this does not intrinsically bother me in the least, we now have a pragmatic interest in converting these people rather than thumbing our noses at them. There's now a chance we can make serious gains in the mainstream business world without compromising our ideals and commitment to technical excellence — so it's time to reposition. We need a new and better label. (Raymond 1998).

They proposed the term "open source" to describe their new business-friendly movement, which was immediately endorsed by Linus Torvalds. Later that year, they created a non-profit group, the Open Source Initiative. They also established a set of principles, and began to certify licenses that conformed to these principles (Raymond, 1999). The initial licenses fell into two groups: the academic research licenses (BSD, MIT) and licenses of the Free Software movement such as the GPL and LGPL (Rosen, 2004).

While some of the initial participants shared Stallman's beliefs, the hallmark of the Open Source movement was its pragmatism.[3] Raymond (1999) outlines key tactics that continue to be used today, including building on the success of Linux, marketing to top corporate decision-makers, focusing on the Fortune 500 and mainstream business press. The plan also included using the Open Source Definition and OSI certification of licenses to control the boundaries of the Open Source movement. Today, this OSI certification defines IP policies that are acceptable within the F/OSS movement, although the Free Software Foundation maintains its own list of approved licenses.

Such pragmatism was vociferously rejected by Stallman as undercutting the moral and ethical foundation of his Free Software movement:

[3] Some would argue that the decisions were driven by self-interest: many of the initial participants were already working for businesses that made money from Linux, and in 1999 Raymond himself would become a director of one of these firms, VA Linux. But in our ongoing research, we have observed both ideology and self interest motivating leaders of both F/OSS movements.

> The main argument for the term "open source software" is that "free software" makes some people uneasy. … [Some developers] figured that by keeping quiet about ethics and freedom, and talking only about the immediate practical benefits of certain free software, they might be able to "sell" the software more effectively to certain users, especially business. The term "open source" is offered as a way of doing more of this — a way to be "more acceptable to business." The views and values of the Open Source movement stem from this decision.
>
> This approach has proved effective, in its own terms. Today many people are switching to free software for purely practical reasons. … Sooner or later these users will be invited to switch back to proprietary software for some practical advantage. Countless companies seek to offer such temptation, and why would users decline? Only if they have learned to value the freedom free software gives them, for its own sake. It is up to us to spread this idea — and in order to do that, we have to talk about freedom (Stallman, 2002)

Comparison of the Open Software Movements

The founding of the Free Software and Open Source movements can be traced directly to the Unix expertise of their respective founders. Both Stallman and Torvalds began their development to make a Unix-compatible OS that fit their requirements. The creators of the Open Source Initative and Apache project all came from a Unix, BSD or Linux background (DiBona et al, 1999; Moore, 2002).

Thus, it is not surprising that many of the successes and guiding principles of the F/OSS movements built directly upon prior successes in the Unix movement. Above all, these movements value free access to source code, which began with AT&T's (paid) licenses for Unix source code. This was cemented by Berkeley's campaign to make source code available to the public at no charge, forming the nucleus of open source projects such as FreeBSD. Among technical users in the Open Standards movement, access to source code was highly valued for the flexibility and control they provide. *The same source code* also reduces vendor lock-in, which during the Open Systems movement was achieved through open standards. And the free source (of BSD, Linux and other F/OSS projects) was enthusiastically sought by students, hobbyists and other individual users (Moore, 2002)

But there is more to the F/OSS movements than IP policies. As West & O'Mahony (2005) note, F/OSS projects are defined not only by an IP policy but a virtually dispersed development methodology and a community governance model. As a development approach, the two F/OSS movements are indistinguishable. They share the same development tools and even websites such as SourceForge.net.

The two movements also share with the earlier BSD movement a technical ethos and pride in technical achievement, in part because many of the authors were writing software for their own use and for approval and adoption by their peers. Often cited is the comment (attributed to Eric Raymond) that open source software is based upon programmers "scratching an itch," but that observation also describes a tradition of programmers building products for themselves and their peers dating back to Ritchie and Thompson's first Unix efforts in 1969.[4]

In many cases, F/OSS packages have succeeded based on good design and implementations by these motivated and knowledgeable developers. For example, both Linux and BSD offered a low-cost Unix implementation for students; Apache filled a key gap in the web server market (West & Dedrick, 2001; Kogut & Metiu, 2001). One key aspect of good design is a modular architecture, which was a major design goal of the Unix operating system (Salus, 1994: 53). Such a modular technical design (adopted by BSD, Linux, Apache and other F/OSS projects) later proved to be an essential prerequisite to decentralized F/OSS development (Baldwin & Clark, 2000, 2003; Kogut & Metiu, 2001). Early proof of this came when the BSD group attracted volunteers to re-implement the Unix operating system (one module at a time), based on only the published interfaces (McKusick, 1999).

Others have argued that F/OSS software is inherently of higher quality, due to the collaborative development approach. The most often cited claim again comes from Raymond (1997), who wrote "Given enough eyeballs, all bugs are shallow." Other potential technical benefits of F/OSS development include feature improvements through user innovation (Franke and von Hippel, 2003), and enabling high quality peer-to-peer support (Lakhani & von Hippel, 2003).

But there are important differences within the Open Software movements, particularly in the rights and responsibilities of IT vendors in their respective communities. Projects like Apache, Mozilla, and Linux (through the Open Source Development Labs) depend heavily on cash and donated labor provided by major IT vendors promoting their own economic interests (such as selling Linux workstations); their respective leaders are pillars of the Open Source movement but have only weak ties to the Free Software

[4] The exact phrase "Every good work of software starts by scratching a developer's personal itch" is found in Raymond's oft-quoted essay, The Cathedral and the Bazaar (e.g. Raymond, 1997; Raymond, 2001: 32). However, a 1994 history of Unix attributes to Berkeley programmer (and sendmail creator) Eric Allman the observation "One general rule of software design is that you should be writing a program that you want to use" (Salus, 1994: 145).

movement. This is exemplified by the paradox that Linux uses a Free Software license (and builds upon many tools developed by Stallman's Project GNU), but is managed by Torvalds and others at OSDL that clearly prefer the ideology (and branding) of the Open Source movement.

Some of the OSDL sponsors also sponsored earlier academic research, such as IBM and DEC (now HP) sponsoring MIT's X11 development. But otherwise, corporate sponsorship makes the Open Source movement seem more like the Open Systems movement than the BSD/MIT efforts from which it gained valuable technology and IP principles. This cultural divide within the Open Software movements — between engineering and business goals — is illustrated by their respective artifacts. The BSD project of the 1980s and Stallman's Project GNU of the 1990s used fanciful animals and T-shirts to appeal to engineers. Meanwhile, the Open Systems and Open Source industries were organized around trade journals and trade shows that provided opportunities for IT vendors to promote their respective wares to potential adopters.

The goal of the Open Source movement in particular was to promote widespread adoption, by consciously attempting to speak the language of potential adopters in business and other organizations. The question we address is how well the movement succeeded in matching its rhetoric to the interests of users, and to what extent the movement's rhetoric and mobilization efforts influence the adoption decisions of potential users.

3. Research Design

We gathered data on organizational adoption of Free and Open Source Software. In designing our study, we identified differences between F/OSS and proprietary software that were likely to be relevant to users. First, adopters have access to the source code due to the nature of the F/OSS licenses, and thus can view and modify the code to meet their own needs. At the same time, access to source code and the availability of free distributions of F/OSS on the Internet makes it difficult for vendors to charge high prices for commercial distributions.

Second, the F/OSS is produced by a loosely affiliated virtual community of software developers, both inside firms and outside them. While support services may be sold by third parties, such software lacks the single point of responsibility provided by a package from a major proprietary vendor, potentially increasing the perceived risk of adoption. Thus, when technology adopters compare an open source

package to a proprietary one, they are likely to see a very different bundle of costs and benefits, as we found in our field study.

Studying Adoption Motivations

We chose to focus on organizational adoption of the best known F/OSS package, the Linux operating system. We focused on its most popular configuration, the Linux on Intel (aka "Lintel") server platform standard, as it competes with Windows, Unix and other proprietary operating systems.[5]

There were two reasons for this decision. First, IT managers enjoyed a wide range of server alternatives. Unlike on the desktop — where one platform has held more than 90% share since 1997, for servers there were three major categories — Unix servers using proprietary RISC-based processors, servers based on Microsoft Windows and commodity Intel-compatible commodity hardware ("Wintel"), and those using Linux (or BSD) using the same commodity hardware.

The server market was also attractive because Lintel server adoption is one of the greatest successes of the F/OSS movements, as measured both by market share and public notice. In 1999, the number of Linux servers passed the number of Unix servers (West & Dedrick 2001) From 1999 to 2002, IDC estimated that annual shipments of new Linux servers increased from 173,000 to 598,000, while revenue from their sales increased from $749 million to $2 billion (Shankland, 2003). Coming in direct competition with Microsoft, Sun, IBM and HP, this success has captured a good deal of attention in both the trade and business press.

At the same time, from our respondents we sought to place the Linux attitudes into a broader context of Open Source attitudes and adoption. As such, we gathered data about additional Open Source server software, of which the Apache web server was the most frequently mentioned.

Data Collection

We used a comparative case study approach, which allows researchers to capture the experience and context of actors as they relate to decision-makers (Benbasat et al. 1987). We conducted a series of in-depth interviews (mostly in person) with U.S. firms and government agencies from November 2002 through October 2004 (Table 3). The semistructured interviews were guided by a common protocol, and

[5] "Unix" and "Open Systems" are often used interchangeably, but we adopt the classification scheme of West (2003), in which proprietary Unix-based RISC systems reflect an intermediate point between fully open and fully proprietary systems.

typically lasted from 45 to 90 minutes. Where possible, we talked with both the CIO (or other senior MIS executive) and an operational staffer such as a system administrator. We hoped that by doing so we could develop a more complete picture, incorporating the view of both top management and those "in the trenches." Also, we could provide a degree of data triangulation (Benbasat, et al, 1987) by comparing the responses of the two interviewees for consistency.

From this data, we used established procedures for generating theory from qualitative data (Glaser and Strauss, 1967; Eisenhardt, 1989; Dubé and Paré, 2003). As recommended by Glaser and Strauss (1967) and Benbasat et al. (1987), site selection and protocol evolved over time. An initial set of interviews was conducted in four organizations chosen on dimensions of theoretical relevance, particularly size and industry. These interviews were transcribed and coded to look for patterns among responses, and a preliminary set of findings was developed. We then continued to refine the protocol with questions to pursue those findings further and to add organizations to the sample to capture additional categories such as technological sophistication and additional industries.

Data analysis was flexible and opportunistic (Benbasat, et al. 1987; Eisenhardt, 1989); it included coding interview notes, developing visual displays of the data on white boards, and counting the number of interviewees who mentioned a particular point. Interviews were also studied and tapes listened to in order to distinguish the emphasis placed on different factors, and to find quotes that were especially striking in representing a particular point of view.

4. Respondents' Perceptions of F/OSS Benefits

After analyzing the findings from the interviews, we next compared the respondents' perceptions of F/OSS to the benefits claimed by the Free Software and Open Source movements. In particular we look at user perceptions of attributes of F/OSS and potential benefits and costs of adoption. These include the importance of access to source code, the importance of cost in motivating adoption, the importance to users of having control over their destiny, and the perceived quality of open source versus closed source software and associated support. Finally, we look at users' perceptions of the movements themselves and the role of movement advocates as change agents within organizations.

Access to source code

Both the Free Software and Open Source movements agree that providing access to source code is critical to enabling users to adapt software to their needs and to share their improvements with the community. This access is undoubtedly vital for those users who want or need to modify the code for their own purposes, or who want to fix bugs that they find. However, our interviews suggest that most IT organizations and professionals are not interested in doing so. Instead, they simply want to install, run and administer software, and have no need or sometimes even ability to analyze and modify the code. A typical view expressed by one interviewee is "Our people are not the type to start tweaking source code or doing their own bug fixes."

Perceptions of the utility of access to source code vary from valuable to dangerous. Most favorable was NatLab, which saw access to source code as vital in its efforts to develop systems capable of highly sophisticated scientific analysis. Biobranch's Informatics Director also saw value, saying, "We knew we had to develop a lot of custom tools, and without having access to the open source community and to the source code, it would be really difficult." Biotech's Associate Director of IT Infrastructure saw both benefits and risks, saying, "It's nice to have, but having source code is like having a gun with bullets. You can shoot yourself in the foot." By contrast, for FastFood, such tinkering was something to be avoided completely, "We don't want to have anybody mucking with that. We're trying to use as much off-the-shelf software as possible." As a measure of the value of source code, eleven of the fourteen organizations interviewed were using Linux is some capacity (and another used Apache), yet only four actually made any use of source code.

The differences in perception might be explained by several factors. First is user experience and skills. E-Store's CTO said his company has some of the world's best Linux kernel engineers, who are fully comfortable modifying the source code if needed to achieve the performance required for its high volume e-commerce business. By contrast, BeachCo's VP of Administration admitted that he could not even administer a Unix system without having a manual nearby, to say nothing of actually looking into the source code.

A second factor is the maturity of the software. Several respondents who did use and even modify source code at times stated that it was not necessary for mature products such as Linux, Sendmail and Apache. BioBranch's Informatics Director stated that "In the olden days, I've recompiled the Linux

kernel and kind of tuned the kernel. But now I don't do any modifying at the kernel level." Biotech's Associate Director of IT Infrastructure said that "Linux is more mature now and doesn't need modification."

A third factor is that users do not see value in modifying underlying infrastructure programs such as Linux and Apache, but are more likely to modify applications that are directly related to their business and customers. An example is E-store, whose CTO said:

> "We do participate with the open source movement, not with the operating system itself, but with other pieces we use. We modify code for applications because it's closer to where our customers are. We're not an operating systems company."

Open source means lower cost

Both the Free Software and Open Source movements have downplayed the importance of cost as a motivation for adopting free or open source software. Among users however, lower cost is by far the most commonly cited benefit of open source software. Everyone who had adopted open source mentioned cost, and in nearly every case cost was the main reason for adopting. For instance the CTO of E-Store, which has made extensive use of Linux and other open source software stated his views on open source in a way that quite closely matches Stallman's warnings about utility-driven adoption:

> "That's our open source policy: if it's cheaper, that's what we do. If some proprietary vendor comes to us and says here's a proprietary solution that has lower cost and higher performance and reliability, then we'll go and do that."

The cost calculation of open source versus proprietary software is more complicated than the "free beer" notion would suggest however. In the case of Linux, a major cost advantage relative to proprietary Unix systems is due to the fact that Linux runs on cheap Intel-based hardware, rather than more expensive proprietary hardware from Sun, HP or IBM. This is very important for organizations that have committed to running some variant of Unix and could do so more cheaply with Linux, as was the case for ISP, E-Store, BioBranch, NewMedia and Dataco. When the alternative under consideration was Windows, the hardware issue was not relevant, as Windows runs on the same hardware platform as Linux. At least some users considered the total cost of ownership, including licensing, support, and internal staff costs, in their calculations. In the case of SouthU, this meant that the beer was not free or necessarily even cheap.

> "It's 'free' — licensed free, but it's not free to use. You guys have heard the saying, "free as in beer"? It's not free as in beer… You have to have the people there to maintain it and develop it and foster it and all those things, and that costs money. And that costs *more* money than the actual licenses for the software."

Still, for some users, the lower direct costs paid for open source software were a factor. This was true especially for businesses that have to pay the commercial price for proprietary software, not so for the universities that enjoy the educational discount. Some, such as ISP, do not even pay the cost of a commercial Linux distribution, preferring to save money by downloading the free version.

Open source means more control and choice for users

Both Free Software and Open Source advocates claim that free or open source software gives users control over their own destinies, and frees them from the tyranny of commercial software vendors ("don't you *want* to be out from under Microsoft's thumb?")

As one manifesto wrote:

> Programming is also about empowerment, what Eric Raymond calls "scratching an itch." Most Open Source projects began with frustration: looking for a tool to do a job and finding none, or finding one that was broken or poorly maintained. Eric Raymond began fetchmail this way; Larry Wall began Perl this way; Linus Torvalds began Linux this way. Empowerment, in many ways, is the most important concept underlying Stallman's motivation for starting the GNU project (DiBona et al, 1999: 13-14).

This is one issue on which many users agreed with the movement's rhetoric and valued the benefit touted by its advocates. Even SouthU, whose CIO had standardized on Windows and was somewhat uncomfortable about open source in general, adopted the open source Apache and ModPerl applications to host a custom-developed course management program when faced with a major price increase from the vendor of a proprietary version of the Perl programming language.

> "(The vendor) had originally established an educational price for the support contract, and they abolished that, which essentially made our contract equal to that of any commercial enterprise. It would have been hard for us to pay, but in addition, if (the vendor's program) fails, (our program) is down and we can't fix it because we didn't write (the vendor's program)."

A more direct critique of closed source vendors was lodged by the CIO of Biotech. His belief that standardizing on Windows made organizations vulnerable to "extortion" on the part of Microsoft had been confirmed in his mind by changes in the vendor's licensing policies just prior to the interview.

> "Microsoft has done such an excellent job of alienating IT people, with their new licensing schemes and their really extortionate attempts to keep their margins up there at historic levels, that everybody's looking for alternatives. Now everyone's feeling the same way because they've been burnt. They've seen what happens when you have a pseudomonopoly situation developing in your company. You shouldn't be surprised when your vendor extracts pseudomonopoly rents."

This view was echoed by NorthU's Associate Director, who said they had adopted Linux partly because of concerns over Microsoft's control. "There was a lot of disenchantment with Microsoft, the monopoly, and being dictated from Redmond. So we were very interested in alternatives."

Another aspect of the choice/control issue was the concern that a proprietary product would disappear from the market, because the vendor either disappeared or decided to abandon that product line. This was expressed by the CIO of SemiCo, who stated, "If the world is moving to a Linux standard, even over a very long time frame, you don't want to be on the wrong path following a proprietary standard."

Open source means better software and support

The primary benefit claimed by the open source movement is that the open source process leads to more reliable software and to faster improvements, adaptations and bug fixes ("astonishing" compared to closed source development). We have no way to empirically measure reliability or other indicators of software quality and compare open to closed source products. We do have strong evidence of users' perceptions of quality, and here there is little support for the idea that open source software is qualitatively better than proprietary software.

Considering Linux in particular, there was a fairly wide range of opinion. Biotech's Associate Director of IT Infrastructure felt it was just as good as, or even better than, the proprietary HP-UX Unix that they also used. FinCo actually tested Linux against Solaris and AIX on equivalent hardware and said that Linux outperformed the others. Others such as SemiCo's CIO were ready to move some applications (such as an SAP module), but not others (a critical database) to Linux, due to reliability concerns. FastFood was only comfortable using Linux for print or file servers, not for any enterprise applications.

When compared to Windows, Linux was found by several to be more reliable and secure, but others found Windows to have the advantage in ease of administration thanks to better tools and user interface.

Only one interviewee, NewMedia's CTO, felt that open source software was inherently inferior, stating "the quality of the code is crappy — you get what you pay for. It's all written by college students." He did admit that a program such as Apache was "good enough for what it does. A lot of stuff that we do on the web isn't like writing shrink-wrap software, and the level of catastrophic failure isn't that catastrophic."

Even E-store, which is the heaviest Linux user among our respondents, admitted that Linux still is not as reliable as the proprietary Tru64 Unix system, and suggested the open source model might be a barrier to reliability. Their CTO stated, "There may be some fundamental issues around the open source model. Is there an incentive to create software that's as reliable as the incentive that an HP would have?"

A contrary opinion was stated by BioBranch, whose Informatics Director felt that the open source process helped support innovation that benefited his company.

> "As an end user, I couldn't care less if Linux is open; we're not going to modify that. But if it was closed, changes to it would be a lot slower. Other people are doing advanced things like parallel computing, and if it weren't open, they wouldn't be making the changes we'd like to see."

While opinions vary about the quality, reliability and speed of innovation in open source software, there is no general perception that the open source model inherently leads to better quality software.

Another issue related to the quality of the software itself is the matter of technical support for users. IT organizations are accustomed to having support from vendors for the software they use, either as part of the cost of the software, or as a separate contract. Since there is no vendor for open source, users must get support either from the open source community via online forums, from vendors of particular distributions, such as Red Hat or Novell, or from third-party service companies that offer support contracts. The open source movement argues that the community provides better support than that offered by vendors (whose support teams it considers almost useless), and says that open source software needs less support anyhow. In its response to the "myth" that it is hard to get reliable support for open source software, OSI claims that "business users will generally find that mature open-source products are far

more reliable to begin with, and that when support is needed it is dramatically cheaper and easier to get than from closed vendors." (http://www.opensource.org/advocacy/faq.php)

This view is quite different from that of our user respondents. Most of them were uncomfortable relying entirely on the community for support, even though several acknowledged that their experience with the community had been quite positive. For instance, SemiCo's CIO said that, "We now have to go through enormous effort to ensure patch compatibility. With Linux you get the latest patches every day." Yet he, along with most others, said that vendor support was either necessary or would greatly increase their comfort level with adopting open source. Of particular importance were support for Linux from a Linux distributor, or from a hardware vendor such as HP or IBM. Most critical was support from application vendors, as no one was willing to run an application on a Linux server without certification and support from the application vendor.

In the end, MIS organizations and professionals are quite risk averse, as the easiest way for them to get noticed by their higher-ups is to have a system fail. In this case, having a support contract means not only having support, but also someone else to hold accountable for resolving the problem. As SouthU's CIO stated,

> "I'm nervous about open source. I'm not paying anybody to support it and thus I'm depending largely on goodwill and luck and skill of my own people affecting their own solutions and soliciting solutions from other people for free. That explanation looks amateurish when you offer it to a dean or to a faculty when a production system is down in my opinion."

Views on the movement and the role of advocates

Our interviewees expressed a range of views on the F/OSS movement itself. A few were strong enough advocates to be considered members of the movement. One was the Web Systems Administrator of NorthU, whose interest in open source began as a student at the same university.

> "When I got my master's in information management, I was working a lot alongside people who do research completely on social issues. I was affected by social impacts of technology coming out of that program. I was concerned about the proprietary software movement versus the open. If I had to say anything, open source fits with my personality. I don't like being subject to the limigtiatons set by someone else."

Others identified themselves as advocates, including respondents at Biotech and FinCo. At the opposite end was the CTO of NewMedia, who claimed he steered clear of hiring "people with GNU disease" and said that, "I don't believe that all software should be free, which is the underlying philosophy of these groups."

The prevailing view was pragmatism. Users select the platform or application that best suits their needs, in terms of price, performance and features. This was summed up by NorthU's CIO, who said, "We're agnostic here. We're not religious about operating system platforms."

Members of a computing movement are potential change agents within organizations. Depending on their position and power, they could do anything from advocating open source software to making organizational policy towards open source. We found that movement believers did attempt to advocate for open source, but did so within the bounds of institutional restraints. For instance, they would support open source options when a new application was being developed or a change of platform was being considered for other reasons, but did not advocate switching existing systems to open source. Also, arguments for open source were made in terms of cost or performance; no advocate would argue that the organization should adopt open source because closed source software is "anti-social" or "tramples on users' freedoms" as suggested by Richard Stallman.

An example of the role of advocacy was Biotech's Associate Director, who stated that he "made sure that we consider Linux whenever an operating decision is made or up for reconsideration." This is confirmed by the company's CIO who pointed to the Associate Director for Infrastructure as having played a role in Biotech's increased use of Linux, and suggested we interview him. A similar role was played by the Web Systems Administrator at NorthU. His supervisor, the Associate Director, said that "I had to be convinced by (the Web Systems Administrator) to bring up (a new application) on Linux." The FinCo advocate said that he and others had pushed Linux use "under the radar" in pilot projects, and later got Linux accepted as a standard within the company. This status was further legitimized by the creation of a formal Linux group within the IS organization.

The role of open source advocates as change agents is clearly constrained by organizational goals and norms, and by their position within the organization. However, the evidence suggests that such advocates can have an impact in encouraging adoption by their organizations.

5. Conclusions

Comparing Motivations of Activists and Beneficiaries

The picture compiled of a computerization movement will depend on the point in time when it's compiled. As has been pointed out by Rogers (1962), the earliest individuals to adopt an innovation are more knowledgeable, information-seeking and risk-taking than the subsequent early and late majorities of the population. So with any innovation adoption, the earliest adopters are atypical of the eventual mass market — they adopt (or join a movement) based on an idea rather than any fully-formed realization of success. The earliest case studies, advocacy reports and academic research on F/OSS adoption (e.g., those with data from 1999 or earlier) would of necessity focus on the earliest adopters, in this case primarily programmers/hackers and users who have bought into the movement's ideology. Meanwhile, our Linux and Apache users show adoption patterns consistent with the early majority, who Rogers (1962) terms "deliberate" and Moore (1991) refers to as "pragmatists".

We also saw evidence of an inherent clash between the hacker culture of F/OSS developers and the corporate or organizational cultures of potential adopters. The rhetorical appeals of the Free Software Foundation in particular were aimed directly at other programmers, but held little appeal for corporate IT buyers and users. In contrast, the founders of the Open Source movement correctly anticipated the pragmatic motivations of these users and have attempted to appeal to their interests. So to what extent do the goals and rhetoric of the movement match the goals and perceptions of the users? We find the following (Table 4).

- The fundamental objectives of organizational users are utilitarian. Our respondents were interested in selecting, deploying and supporting cost-effective software that satisfied key requirements for features and reliability. This is consistent with the tone of the Open Source movement, which claims it just wants to promote open source software "on solid pragmatic grounds rather than ideological tub-thumping." On the other hand, the Free Software

movement criticizes the Open Source movement for appealing to "narrowly practical values," i.e., the very things that matter to users.

- The lynchpin of both movements — full access to software source code — was surprisingly of mixed value to users, including even some of the more aggressive open source adopters. Some firms valued having source code — usually for rapidly evolving business-driven applications, rather than stable infrastructure software. In other cases, the respondents identified the availability of the source code as "nice to have" but said that it had little utility for them. And in a few cases, respondents worried that the ability to modify source code could ruin configuration control efforts, increasing future support costs and overall system risks.
- A key argument of the Open Source movement—that the open source process leads to more reliable software, faster development time, and better support via the online community—was clearly not consistent with the perceptions of users. While there were a range of views, most respondents felt that Linux was somewhat less reliable than proprietary Unix operating systems, while more reliable but harder to use than Windows. And while the support of the open source community was often praised, very few users were willing to forego the security of a formal support contract from a reputable IT company.
- One point of agreement betweeen the movements and our respondents was the value placed on having more control over one's destiny and not being at the mercy of proprietary software vendors. Open software is perceived as increasing user choice and control, which has both ideological value to the movements and pragmatic value to users.
- The most important benefit of open software to users is the one that the movements both try to downplay, i.e., low cost. While Stallman commands us to think "free speech, not free beer," and the open source people emphasize quality rather than cost, it is clear that organizational users are thinking about cost more than anything else when considering F/OSS. In fact, some users saw the somewhat utopian open source development process mainly as way of getting volunteers to develop software at a low cost: as one stated, "we want to be free riders." Others mostly appreciated the fact that Linux offered them a way to run a Unix quality operating system on cheap Intel hardware. This may not be surprising,

given that IT organizations operate under tight budgets (especially in the post dot.com era), but it does present a dilemma for the movements. It might reduce the motivation of top programmers to volunteer their time if their efforts are being touted as mainly a way for user organizations to save money on software.

- Finally, we found that most users were agnostic about open versus closed source software, with no one calling for their organizations to refuse to deal with vendors who refuse to reveal their source code (as both free and open source rhetoric advocates). We did find a few movement members operating within the user organizations, and in several cases they appeared to be acting as change agents promoting adoption. Their advocacy was constrained by the goals of the organization, so they tended to argue for open source on utilitarian ground. As such, they might be seen as ideologically impure by the Free Software movement, but in their own way were actually carrying out the objective of the Open Source movement to promote open software on pragmatic rather than ideological terms.

6. References

Baldwin, Carliss Y. and Kim B. Clark, "The Architecture of Cooperation: How Code Architecture Mitigates Free Riding in the Open Source Development Model ," HBS - MIT Sloan Free/Open Source Software Conference: New Models of Software Development, Cambridge, Mass., June 2003, URL: http://opensource.mit.edu/papers/baldwinclark.pdf

Baldwin, Carliss Y. and Kim B. Clark, *Design Rules,* 2000

Benbasat, Izak, David K. Goldstein and Melissa Mead, "The Case Research Strategy in Studies of Information Systems," *MIS Quarterly* (11:3), Sept. 1987, pp. 369-386.

DiBona, Chris, Sam Ockman and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution,* Sebastopol, Calif.: O'Reilly, 1999.

Dubé, Line and Guy Paré, "Rigor in Information Systems Positivist Case Research: Current Practices, Trends, and Recommendations," *MIS Quarterly* (27:4), Dec. 2003, pp. 597-635.

Eisenhardt, Kathleen M., "Building Theories from Case Study Research," *Academy of Management Review* 14, 4 (Oct. 1989): 532-550.

Elliott, Margaret S. and Walt Scacchi (2004). "Mobilization of software developers: the free software movement. To appear in *Information, Technology and People, Festchrift for Rob Kling.*

N Franke, N and E von Hippel, (2003) Satisfying Heterogeneous User Needs via Innovation Toolkits," *Research Policy*, 32, 1199-1215.

Free Software Foundation, "The Free Software Definition," revised 2005/02/24, URL: 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.

Glaser, Barney G., and Anselm Strauss, *The Discovery of Grounded Theory: Strategies of Qualitative Research*. London: Wiedenfeld and Nicholson, 1967.

Grindley, Peter, "Open Computer Systems: A Standards Revolution," in *Standards Strategy and Policy: Cases and Stories,* Oxford: Oxford University Press, 1995, pp. 156-194.

Iacono, Suzi and Rob Kling, (2001). "Computerization Movements: The Rise of the Internet and Distant Forms of Work, in Yates, J.A. and J.V. Maanen (Eds.), *Information Technology and Organizational Transformation: History, Rhetoric and Practice*. Sage Publications.

Isaak, Jim; "POSIX Inside: a Case Study," HICSS38, IEEE Computer Society Press, CDROM, 10 pages, Jan 2005

KernelTrap.org, "Linux: Open Source Ideology," April 21, 2002, URL: http://kerneltrap.org/node/159, accessed 2/25/2005.

Kling, Rob and Suzi Iacono (1988). "The Mobilization of Support for Computerization: The Role of Computerization Movements," Social Problems, 35(3), 226-242.

Kogut, B and A Metiu Open Source Software Development and Distributed Innovation Oxford Review of Economic Policy, 2001

Lakhani, K and E von Hippel, (2003) "How Open Source software works:'Free' user-to-user assistance," *Research Policy* 32,7 (2003): xxx-xx.

Lerner, Josh and Jean Tirole, "Some Simple Economics of Open Source," *Journal of Industrial Economics,* 52, 2 (June 2002): pp. 197-234.

Markus, M. LL, Manville, B. & Agres, C. E. (2000). What makes a virtual organization work? *Sloan Management Review* 42(1): 13-26.

McKusick, Marshall Kirk, "Twenty Years of Berkeley Unix," in DiBona, Chris, Sam Ockman and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution,* Sebastopol, Calif.: O'Reilly, 1999, 31-46.

Moody, Glyn, (2001) *Rebel Code: Inside Linux and the Open Source Revolution,* Cambridge, Mass.: Perseus.

Moore, J.T.S., *Revolution OS,* movie, 2002.?

Morin, Richard, "BSD-based OS releases, part II, UNIX Review, Vol. 13, No. 4, April 1995.

Open Source Initiative, "Frequently Asked Questions," URL: http://www.opensource.org/advocacy/faq.php, accessed 2/25/2005.

Open Source Initiative, "The Open Source Definition," Version 1.9, 2002 URL: http://www.opensource.org/docs/definition.php accessed 2/25/2005.

Raymond, Eric (2001). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*, O'Reilly, Sebastopol, CA

Raymond, Eric S. (1997). "The Cathedral and the Bazaar," June 23, 1997, URL: http://hackvan.com/pub/stig/articles/cathedral-paper/cathedral.txt, accessed 2/25/2005.

Raymond, Eric S., Goodbye, "free software"; hello, "open source"., Feb. 8, 1998, URL: http://www.catb.org/~esr/open-source.html

Raymond, Eric, "The Revenge of the Hackers," in Chris DiBona, Sam Ockman and Mark Stone, eds., *Open Sources: Voices from the Open Source Revolution,* Sebastopol, Calif.: O'Reilly, 1999, pp. 207-219.

Rogers, Everett M., *Diffusion of innovations,* New York: Free Press, 1962.

Rosen, Lawrence (2004, *Open Source Licensing: Software Freedom and Intellectual Property Law,* Upper Saddle River, NJ: Prentice Hall PTR.

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

Shankland, Stephen, "IDC: Servers to Make Mild Recovery," CNET News.com, May 23, 2003, URL: http://news.com.com/2100-1010_3-1009814.html

Stallman, Richard M. (2002). *Free Software, Free Society: Selected Essays of Richard M. Stallman*, GNU Press.

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.

West, Joel and Siobhán O'Mahony, "Contrasting Community Building in Sponsored and Community Founded Open Source Projects," Proceedings of the 38th Annual Hawai'i International Conference on System Sciences, Waikoloa, Hawaii (Jan. 2005), URL: http://opensource.mit.edu/papers/westomahony.pdf

West, Joel "Understanding Open Source Licensing: Three How-to Guides," *IEEE Software* 22, 3 (May/June 2005).

West, Joel, "How Open is Open Enough? Melding Proprietary and Open Source Platform Strategies," *Research Policy* 32,7 (2003): 1259-1285.

West, Joel, and Jason Dedrick, " Proprietary vs. Open Standards in the Network Era: An Examination of the Linux Phenomenon," Proceedings of the 34th Annual Hawai'i International Conference on System Sciences, Maui, Hawaii (Jan. 2001): 5011.

7. Tables and Figures

Movement	**Unix**		**Free Software**	**Open Source**
Starting date	1969		1984	1998
Submovement	**BSD**	**Open Systems**		
Leadership orientation	Technical excellence	Business adoption	Technical excellence	Business adoption
Philosophical Goals	Academic research	Vendor independence	Software freedom	Pragmatic success
Licenses*	MIT (1985) BSD (1989)	*proprietary*	GPL (1989) LGPL (1991)	MPL (1998?) CPL (2001)
Key technologies	V7 (1979) BSD (1977) X11 (1987) Motif (1990) 4.4BSD-Lite (1994)		GNU Emacs (1984) gcc (1987) Gnome (1997)	Apache (1995) MySQL (1995)
Key Institutions	IEEE POSIX (1984) X/Open (1984) Open Software Foundation (1988) Unix International (1988) Open Group (1996)		Free Software Foundation (1985) Software Freedom Law Center (2005)	Open Source Initiative (1998) Apache Software Foundation (1999) Open Source Development Labs (2000)
Sponsors	ATT, DARPA, DEC, IBM, Sun			IBM, HP, Intel
Spokesmen	Bill Joy		Richard Stallman	Linus Torvalds Eric Raymond

* Licenses conforming to the Open Source Definition and approved by the Open Source Initiative

Table 1: Contrasting the Open Software movements

Free Software Foundation *Four Freedoms*	**Open Source Initiative** *Open Source Definition*
0. The freedom to run the program, for any purpose. 1. The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this. 2. The freedom to redistribute copies so you can help your neighbor. 3. The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. Access to the source code is a precondition for this.	1. Free Redistribution [*without restriction or royalty*] 2. Source Code [*must be included*] 3. Derived Works [*must be allowed and distributable]* 4. Integrity of The Author's Source Code [*may be protected*] 5. No Discrimination Against Persons or Groups 6. No Discrimination Against Fields of Endeavor 7. Distribution of License 8. License Must Not Be Specific to a Product [*so technology can be used in other products*] 9. License Must Not Restrict Other Software [*allowing mixing with proprietary software*] 10. License Must Be Technology-Neutral

Source: Free Software Foundation (2005), Open Source Initiative (2002)

Table 2: Basic principles of the free and open source software movements

Name	Business	Org. (unit) Size†	Primary Platform	Linux Adoption	Informants
Beach Co.	Rec. equipment	80	Windows	Website only	1
Bio Branch	Pharmaceuticals	560 (150)	Linux	Predominant	1
Biotech	Pharmaceuticals	1,000	Unix	Internet and database applications	2
Dataco	Online data retrieval	2,700 (1,500)	Linux	Phasing out Unix	1
E-store	E-commerce	7,500	Unix	Shifting from Unix to Linux	1
FastFood	Restaurant chain	200,000	Mixed	None	1
FinCo	Financial services	130,000	Mixed	Partial adoption	1
NatLab	Government research lab	(8,000)	Unix	Phasing out Unix	2
ISP	Internet service provider	11	Linux	Since founding	1
NewMedia	Content provider	35	Unix	Partial transition	2
NorthU	Public university professional school	114,000 (325)	Mixed	Replacing Unix with Linux, while keeping Windows	3
Semico	Semiconductor design	2,500	Mixed	Limited; evaluating further use	2
SouthU	Public university professional school	114,000 (300)	Windows	Abandoned previous limited use	2
Travel Service	Travel-related reservations	6,000	Mainframe	Partial adoption	1

Total: 14 companies, 21 informants

† Size of parent organization (unit) in number of employees

Table 3: Characteristics of sample firms

	Free software movement	**Open source movement**	**Organizational adopters**
Primary Goal	"Software freedom"	Widespread adoption	Economic utility
Access to source code	Fundamental	Fundamental	Valued by some, used selectively
Better quality software and support	Not emphasized	Primary benefit of open vs. closed source	Mixed views on quality, want a support contract
Control and choice	Key goal	Key goal	Important to many
Cheap software	"Think free speech, not free beer"	Not emphasized	Key benefit, universally valued
Perceptions of the movement	True believers, purists	Pragmatic evangelists	Agnostic

Table 4: Perceptions of movements versus users