THE OPEN SOURCE MOVEMENT: A REVOLUTION IN SOFTWARE DEVELOPMENT

Kevin Carillo

John Molson School of Business

Concordia University, Montréal

kev@biskinito.net

Chitu Okoli*

John Molson School of Business

Concordia University, Montréal

Chitu.Okoli@concordia.ca

Phone: +1 (514) 993-6648 or +1 (714) 724-5049

THE OPEN SOURCE MOVEMENT: A REVOLUTION IN SOFTWARE DEVELOPMENT

ABSTRACT

The open source movement is based on a radical retake on copyright law to create high quality software whose use and development are guaranteed to the public. In this article we trace the history of the movement, highlighting its interaction with intellectual property law. The movement has spawned open source software (OSS) communities where developers and users meet to create software that meets their needs. We discuss the demographic profile of OSS participants, their ideology, their motivations, and the process of OSS development. Then we examine the impacts of OSS on society as a whole from the perspective of the information society, discussing the effects on OSS developers, users of OSS, and society at large, particularly in developing countries.

Keywords: open source software; virtual communities; social impacts of computing; software development methodologies; software engineering

INTRODUCTION

Open source software (OSS) communities are an important type of virtual community today, where members convene online with the common goal of producing software that is valuable both to developers and for the general public using the open source software development (OSSD) methodology. This movement of 20 years or so has drawn increasing attention in organizations of all sizes—commercial, non-commercial, and governmental—and even with individual consumers as high-quality consumer-oriented OSS products have emerged. OSS communities have distinctive cultural artifacts, including community norms, values, and

beliefs. As the open source movement continues to grow both as a software development methodology and as a philosophical/social/political approach to intellectual property, OSS communities are having an increasingly important role in the software industry, and on the society at large that uses software.

This article discusses various important social aspects of the OSS movement, mainly at the level of OSS communities and societies. We begin by briefly tracing the history of the movement, explaining the evolution of the concepts of "free" and "open source" software. In the following section we discuss OSS communities as a type of virtual community. In the final section we discuss the impacts of OSS on society as a whole from the perspective of the information society.

A BRIEF REVOLUTIONARY HISTORY: FROM FREEING SOFTWARE TO OPENING THE SOURCE

> The advent of the Internet is driving some drastic changes in the software industry. We usually think of dramatically lower communications and transaction costs as the Internet's major value […] but there's something else going on, as well. The Internet's engineering tradition, its native culture, and even its folklore are turning out to hold lessons that are going to be critical for the creativity- and software-intensive economy of the coming century. [36]

Free software (later renamed "open source software") appeared even before people started thinking in terms of proprietary software, at a time when software development was ruled by open source principles [47]. In the 1960s and 1970s, software programming was mainly performed in both academic and corporate laboratories by scientists and engineers who freely gave, exchanged and modified software. In the early 80s, as software programming increasingly turned proprietary, Richard Stallman founded the Free Software Foundation (FSF) to define and

diffuse legal mechanisms and conceptual principles of what he called “free software” [19, 49]. By writing the GNU Manifesto (Stallman, 1985), he communicated his ideological view of the nature of intellectual property rights as regards software, and started attracting convinced developers to join him in his GNU Project (GNU stands for “GNU’s not UNIX”). In 1989, the FSF released the GNU General Public License (GPL) version 1 (the updated version 2 was released in 1991) which legalizes “copyleft” mechanisms and grants end-users freedoms in software copies and derivative works.

Turning copyright around

“Copyleft” as expressed by the GPL has had a critical effect on shaping the very existence of open source software communities. Open source software uses copyright law to preserve certain freedoms (hence the name, “free software”) regarding the creation, modification, and sharing of software. Specifically, all open source software grants users the following key rights:

1. **The right to full access to the source code.** When a computer programmer sees how a piece of software actually works, as specified in the source code, they can fully understand the inner workings and can intelligently modify the software as they deem appropriate.
2. **The right for anyone to run the program for any purpose without restriction.** There are no restrictions against commercial, military, foreign, or any other use, and discrimination against users for any reason is expressly forbidden.
3. **The right to modify the source code.** This includes absorbing the software, in whole or in part, into other pieces of software created by other developers.
4. **The right to distribute both the original software and the modified software.** A key difference between “free software” and “freeware” is that while freeware generally permits

and encourages free distribution of the software, it does not permit sale of the distributed software beyond reasonable distribution costs; free software, in contrast, permits resale at any price.

5. **The right to know about their open source rights:** The open source license must be prominently displayed and distributed to users, so that they are aware of their rights (including access to the source code). Practically, since users are aware that they can obtain the source code for free, the sale price of OSS tends to be zero, or quite low.

While the preceding five rights constitute open source software [32], the FSF's GPL, the first legal document to license open source software, goes further. The GPL grants users and developers these rights with the intention that developers would modify the software and share it with others with similar liberality, and in accordance with Stallman's personal beliefs on the ethical rightness of sharing software, the GPL assures sharing by further incorporating the concept of "copyleft". Copyleft is an obligation that the distributor of OSS agrees to in order to receive the privileges mentioned above:

6. **The obligation to distribute derivatives under copyleft**. Any software modified under the GPL can be redistributed for sale, but it must be licensed under a copyleft license; that is, modified derivative works must also be made available under an open source license. While it does not have to be licensed under the GPL itself, the chosen distribution license may not restrict any of the five rights listed above.

These copyleft terms are critical to the very existence of OSS communities. When Richard Stallman posted his manifesto and invited software developers to join him in his crusade for free software, there was no lack of sympathetic and willing hackers who wanted a return to the days of free sharing. However, there was a grave concern that, corporate interests could

easily take these programs, add their proprietary extensions, and withdraw the software from public access. With its copyleft mechanism, the GPL guaranteed that any person or corporation who wanted to benefit from the liberal efforts of computer programmers would be legally bound to share their work in the same spirit of camaraderie. Considering the climate in which the free software movement was founded, it is unlikely that the movement could have gotten off the ground without such a radical clarion call to mobilize devoted followers in the first place.

From free software to open source software: From ideology to $$$

In 1991, Linus Torvalds, a 21-year-old Finnish programmer, started the famous Linux Project, released under the GPL, which implements a Unix-like operating system on Intel-based microcomputers. The project has grown rapidly to produce a powerful, fast, efficient, stable, reliable, and scalable operating system. The number of Linux users and developers has expanded rapidly. Not including free downloads and installations of Linux, "25% of servers and 2.8% of desktop computers ran [paid distributions of] Linux as of 2002. The Linux market is rapidly growing and is projected to exceed $35.7 billion by 2008" ["Linux", 50]. Moreover, version 2.2.10 of the kernel lists 190 names, though the total number of contributors was estimated at around 1,200 [9, 41].

A major paradigm shift occurred in 1998, when Bruce Perens and Eric Raymond expressed their suspicion that firms may not be convinced by GPL due to Stallman's term, "free" software, "which might understandably have an ominous ring to the ears of business people" [47, p. 1551]. The "open source" software movement was created based on the integrating of all the previous licensing that had been prior designed [48]. As a result, a major chasm appeared between Stallman's free software view, which was more ideological and philosophical, and Perens/Raymond's open source view, whose purpose was more business-oriented. The latter was

aimed at fighting against the evil “proprietary software” by convincing firms to rely on OSS.

Several major OSS projects have marked people’s mind in such software revolution. The Apache web-server project started in early 1995 and has become the most popular Web server software, controlling over 60 percent of the market, much more than Microsoft and Netscape both combined [11, 46]. Inspired by Eric Raymond’s paper “The Cathedral and the Bazaar” [35], Netscape, one of the main actors in the Internet browser industry, decided to release its source code by creating Mozilla, its first open source version of its former Communicator. In November 2004, the Mozilla Foundation announced the worldwide availability of the Mozilla Firefox 1.0 web browser.

Today, all major hardware and software vendors have at least forayed into the open source approach. For example, Apple Computer surprisingly followed this model in 2000 when they released the kernel of their Unix-based Mac OS X operating system to the open source community as Darwin 1.0 ["Apple Darwin", 50]. (Only the kernel was released as open source; the signature Aqua graphical user interface remains proprietary.) In 2005, IBM announced their plan to spend $100 million over the next three years to build support for Linux into desktop applications for its Workplace software [51]. Meanwhile, IBM has planned to spread Linux worldwide. In 2004, IBM concentrated on implementing Linux in Brazil, Russia, India and China. Since 2005, the company has planned to increase its efforts in those countries but will also begin extend its programs in Eastern Europe[12].

LIBERTY, EQUALITY, FRATERNITY: OPEN SOURCE COMMUNITIES

> No quiet, reverent cathedral-building here—rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches … out of which a coherent and stable system seemingly emerges only by a succession of miracles. [35]

Open source software development communities have been defined as groups of loosely connected programmers, who use the Internet as a medium for collaboratively developing, improving, and disseminating software [31]. In this section, we discuss OSS communities as a particular genre of virtual community [8, 10, 24], examining the profiles of participants, their ideologies, motivations, and the outcomes of their participation in these communities.

Participants' profiles

It has long been questioned whether OSS participants fit a particular profile, or to what extent OSS developers are representative of the overall population. Recent research has dispelled the stereotype of OSS developers as anarchistic hackers operating on the fringes of society [6, 14]. Several surveys have identified consistent traits among OSS developers. They are almost exclusively male, with an average age of around 27. 20% are students, and over 50% have IT-related jobs; 16% are paid to participate in open source projects (either as part or all of their work responsibilities). When considered that around 70% have at least a bachelor's degree and 20% are students, it is likely that at least 80% have at least some university-level education. Table 1 summarizes findings of some key studies.

The studies have been inconsistent regarding the extent of participation in OSS projects [16, 19, 25]. However, one common finding is that OS developers generally participate in a limited number of projects (from two to three on average). Similar results were found when analyzing a repository of Linux materials containing 4,500 contributor-generated metadata files [9]. However, the studies are inconsistent as at the amount of time spent for OS development (Table 1).

Table 1: Demographic Variables

	Hars and Ou, 2001	Lakhani and Wolf, 2002	Ghosh et al., 2002
Sample	389 developers from various OSS communities	432 developers, from several projects on SourceForge.net	2784 developers from various OSS communities
Survey Date	2000	Fall 2001	February 2002
Gender Proportion	95% male	97.5% male	98.9% male
Age	20-29: 50% 30-39: 35%	Average: 30	Average: 27.1
Geographical distribution	**N/A**	**US-Canada**: 45%: **Western Europe**: 38%	**Western Europe**: ≈ 60% **US-Canada**: 14%
Educational background	**College**: 48% **Master**: 21% **PhD**: 3%	**N/A**	**College**: 33% **Master**: 28% **PhD**: 9%
Professional situation	**Students**: 14% **Paid for OS dev**: 6% **Salaried**: 33%	**Students**: 19.5% **IT jobs**: 58%	**Students**: 20.9% **IT jobs**: 58% (33% - software engineers)
Number of OS Projects	**Current**: 1: 35% 2-4: 47% 5-10: 14%	**Current**: Average = 2.63 (std=2.14) **Total ever worked on**: Average= 4.95 (std=4.04)	**Current**: 1: 28.6% 2: 26.7% 3: 15.4% 4-5: 14.9% **Total ever worked on**: 1-5: 71.9% 6-10: 18.7% 11-20: 6.3%
Time spent per week	**N/A**	Average= 14.3 hours (std=15.7)	**Less than 2 hours:** 22.5% **2-5 hours:** 26.1% **6-10 hours:** 20.9% **11-20 hours:** 14.3%

OSS communities consist not only of software developers, but also include non-developer users, who reflect a broader demographic spectrum [15]. In each OSS project, members have to fulfill specific tasks that are integrated in the holistic development process. As a consequence, despite the common view of OS participants as willing, independent enthusiasts, deeper analysis reveals hierarchies in the communities [17]. Non-developers are responsible for dealing with bug and enhancement reporting, and end-user documentation writing, whereas

developers update, modify the source code and make important decisions concerning further development.

Ideology and values

As members of OSS communities, OS participants share common interests and goals. One of the main emphases in OSS communities is the social interaction among participants through electronic communication. Howard [21] defines virtual communities as "social aggregations that emerge from the Net when enough people carry on public discussions long enough, with sufficient human feeling, to form webs of personal relationships in cyberspace", emphasizing the creation of relationships among members.

The most commonly accepted view of the OSS culture has been depicted by Eric Raymond in his famous paper "The Cathedral and the Bazaar" [35]—commonly regarded as the manifesto of the open source movement—characterizing the OSS movement by its gift culture as opposed to the predominant Western exchange culture in which people interact according to their personal interests. Raymond argues exchange cultures rely on *scarcity* whereas gift cultures are based on *abundance*. A gift culture determines social status "not by what you control but by what you give away". Such altruistic values have been empirically studied and demonstrated to promote the creation and maintenance of social relationships in virtual environments [2].

Another factor that may differentiate some OSS communities with others is the degree of hostility towards proprietary software [28]. Such hostility varies from severely hostile—with Microsoft Corporation typically depicted as the leader of the evil empire—to a more tolerant view that defends the acceptance of the symbiotic nature of open source and proprietary software [33]. Table 2 summarizes some of the more predominant cultural values seen in OSS communities [42].

Table 2: Ideologies and values of open source software communities [42]

Norms	• Taboo against forking projects (that is, starting a rebel, though legal, project from the same code base) • Distributing changes without cooperation of moderators frowned on • Removing a person's name from project history, credits or maintainers list is not done without explicit consent
Values	• The best craftsmanship wins • All information should be free • You don't become a hacker by calling yourself a hacker - you become a hacker when other hackers call you a hacker • Non-trivial extensions of function are better than low-level patches and debugging • Work that makes it into a big distribution is better than work that does not
Beliefs	• With enough eyeballs all bugs are shallow • Practice is better than theory
Ideologues	• Richard Stallman (Free Software Foundation) • Eric Raymond (Open Source Initiative)
Language, Symbols	• "Distros" (Linux distributions; that is, customized packagings of Linux) • Free Software Foundation • Copyleft • Open source licenses
Narratives	• The Halloween Papers • The Cathedral and the Bazaar • Slashdot, Freshmeat, Sourceforge

Motivations

Several studies have investigated the different motives for programmers and software companies to produce, license, and use open source software [1]. As Lerner and Tirole [27] asked, "Why should thousands of top-notch programmers contribute freely to the provision of a public good?" Most OSS developers are not paid for their contributions, and the licenses and hacker practices make it very difficult for these developers to appropriate any return from their products [48].

Two main approaches have been adopted by the literature in order to tackle the OSS motivational issue: economic and social. First, the economic perspective posits that a person will contribute to an OSS development project only if the benefits of the contributions will outweigh its costs [18] which signifies that a contribution needs to generate a "net benefit" that consists of the sum of immediate payoff and delayed payoff [27]. Lerner and Tirole identified both the increase in personal use-value of a product and the satisfaction of having achieved something valuable as the two main immediate benefits. Additionally, the career future incentive (future jobs, shares and access to capital market) and the ego gratification incentive (stemming from a desire of peer recognition) are considered as the two main delayed benefits. Moreover, the economic perspective posits that a programmer will use his contributions to signal his skills and knowledge to both the open source community and to the labor market, in line with the signaling incentive perspective of economic theory [20].

Second, Eric Raymond, a leading OSS evangelist, has proposed psychosocial and cultural reasons to explain participants' motivations. Through the redefinition of intellectual ownership, social status in OSS communities is not determined by what you own but rather by what you give away [18]. In Raymond's [35] characterization of the OSS gift culture, reputation (built from the amount of contributions) and peer recognition mainly motivate people to freely and willingly participate to OSS projects.

Table 3: Main motives among OSS participants

Category		Motive	Theoretical			Empirical (rankings)		
			Feller and Fitzgerald, 2000	Raymond, 2001	Lerner and Tirole, 2002	Ghosh et al., 2002	Lakhani and Wolf, 2002	Hars and Ou, 2001
Individual	Economic	To gain future career benefits	X		X	8	**3**	5
		To improve coding skills	X		X	**1**		**1**
		To "strike it rich" through stock options, etc.	X					
		Low opportunity cost—nothing to lose	X					
		Distribute not marketable software products				12		
		Make money				13		8
	Socio-political	Ego gratification and signaling incentives	X		X			
		Sense of belonging to a community	X					6
		Altruism	X			**2**		7
		Participate in a new form of cooperation				4		
		Participate in the OS/FS scene				6		
		Feel personal obligation to contribute because use F/OSS					5	
		Like working with this development team					6	
		Peer recognition		X				**3**
		Get a reputation in OS/FS community		X		11	8	
	Techno-logical	Intrinsic motivation of coding	X				**2**	**2**
		To meet a personal technological need	X			7	**1**	4
		To exploit the efficiency of peer review	X			9		
		To work with "bleeding edge" technology	X					
Community	Economic	To exploit investor infatuation with OSS	X		X			
		To shift from software as a commodity to software as a consumer-driven service	X					
		To raise mind share and strengthen brand	X					
		To exploit indirect revenue	X					
		To make software affordable in developing countries	X					
		To cut costs	X					
	Socio-political	Social movements require an enemy e.g. Microsoft	X					
		Overcomes "digital divide"	X					
		Ideology—software must be free	X			5	**3**	
		Model for wider domain	X					
		Limit the power of large software companies				10	7	
	Technological	To address the software crisis, particular poor quality	X					
		To share tedious development tasks with users	X					
		To leverage the OSS community for R&D	X					
		To promote innovation	X					
		To ensure transparency of the application	X					
		Improve OS/FS products of other developers				**3**		

Feller and Fitzgerald [13] have proposed a much broader view of OS participation motives by striving to encompass all the different factors that had been found by the literature. They have designed a framework that proposes a broad motivational model based on three areas: technological, socio-political and economic. Each area is then subdivided into two factor groups: individual and organization/community factors. This framework has been used in order to integrate the motives that the literature has identified so far from both a theoretical and empirical perspective. Table 3 summarizes the findings of the leading theoretical and empirical literature on motivations for participation [13, 16, 19, 25, 27, 35].

OSS development practices and outcomes

Because OSS is freely accessible to all, OSS development practices differ greatly from the traditional practices of commercial software [46]. Typically, an OSS project begins by trying to solve a user's particular problem [31] that may not have any commercial software alternative, or else an expensive one [22]. According to Raymond [35], "Every good work of software starts by scratching a developer's personal itch. … Too often software developers spend their days grinding away for pay at programs they neither need nor love. But not in the [open source] world …."

The originating programmer realizes that giving away their work to a networked community would encourage valuable contributions from like-minded programmers who find the problem interesting and worthwhile, thus permitting the problem to be solved much faster and more effectively. The early contributors to an open-source project tend to resemble the original project creator: sophisticated programmers, who aim at solving a specific problem [33]. Once an initial version has been programmed, the source code is then released and freely available to all through downloading from a website that has been specifically designed for this project. The

project initiator typically sets up mailing lists for interested people to facilitate interaction for developing solutions, providing information and discussing any critical source code modification [47]. In general, the initiator of an OSS project leads and manages development. Projects usually employ a modular architecture [31] which allows developers to change functionalities without having to modify the core components. Each module is then handled by either an individual or a development group, and all are coordinated through lead architects [38].

It is claimed that the evolutionary process followed by OSS development projects produces better software than the traditional closed process [28, 40] and at lower cost, though this perception has been questioned [3, 4]. However, it is argued that the quality of open source software does not come from its management but from its openness. Free access to the source code [44] ensures that the code is tested and retested by a worldwide user pool, which leads to a higher reliability through a timely identification of bugs and opportunities for software enhancements [30]. However, OSS development certainly does not always lead to high-quality products. Most OS projects do not even reach their final implementation stage, but only the successful examples are typically trumpeted [43]—of course, this is the same with any software development methodology. While not a panacea, OSS development is a practical way to tackle escalating error rates [37]. It seems that OSS high quality benefits from the "many eye balls" approach that enables a broad and deep field testing [35] through both parallel programming and extended review efforts [18].

A NEW REPUBLIC: OPEN SOURCE SOFTWARE IN SOCIETY

The open source phenomenon is often described as a "revolution" and a "movement" because it is far more than merely a radical new approach to software development—it has

significant effects on social structures related to software creation and use at all levels of society in the contemporary information age. In this section we discuss some of these effects, both positive and negative, on software developers, users, and society as a whole.

Effects on developers of open source software

OSSD is primarily a developer-driven movement, where software developers have formulated new structures for themselves in their communal craft of producing software. It began on the individual level, with certain programmers who created legal structures to preserve the benefits of communal software production.

Individual developers

When considering the effects on individual developers, we discuss here the effects on software developers who execute personal projects or projects of which they are the owner, as opposed to an employee or contractee. Most OSS projects begin as a programmer's attempt to "scratch a personal itch" [35]. That is, they have a personal programmatic need, and begin a project to meet that need. Normally, their needs might be larger than what one programmer could easily hack together, especially if it carries no promise of financial remuneration, but with the OSS model they can leverage the skills and labor of several other programmers who share the same "itch", and thus the otherwise niche project becomes feasible. One example of this phenomenon is in the proliferation of open source content management systems (such as Drupal, Joomla!, and WordPress) which are pre-packaged websites. Rather than web developers creating their own administrative interface and scripts, they have converged to develop systems that serve their general needs in website development.

OSS projects provide a unique learning environment by placing contributors into a pool of skilled programmers—the pool is potentially as large as the Internet itself—who help each

other improve their professional skills as they work together [46]. Particularly for less experienced programmers, OSS projects provide an unparalleled opportunity to view the full source of highly professional, high quality major projects. This provides a unique learning opportunity working with live software, and supported with a pool of generally friendly, experienced programmers willing to help them understand the inner workings of the code.

For the more highly-skilled contributors to a project, OSS participation provides a forum to stage their talent before their peers and build up a reputation, in a highly meritocratic social milieu. This peer esteem can also have personal economic benefits, as participation in OSS projects provides ample public evidence of a developer's skills when courting contracts or employment [5, 29, 52].

Organizational developers

Here we discuss the effects of the OSS movement on organizations (commercial, not-profit, governmental and otherwise) that pay developers to participate in OSS projects, whether or not the organization is the owner or sponsor of the project. The OSS model provides an opportunity for them to share financial costs and development effort for large projects for applications that are common across many organizations [18]. This model is particularly suitable for infrastructural software such as operating systems (Linux), web servers (Apache), database management systems (MySQL, PostgreSQL, SQLite), and web-based scripting languages (PHP, Perl, Python). The OSS community gives these organizations access to a larger pool of talent than they would have if they relied only on the developers that they themselves paid. Moreover, the fact that most of the developers are usually from outside the organization diversifies the approach to problem-solving, leading to more versatile solutions. Moreover, once an OSS project has reached its "cruising speed", development and debugging times are significantly shortened.

From another perspective, for organizations where software creation is a goal in itself rather than a means to a goal, the OSS movement creates valuable new business opportunities [46]. By the very nature of open licensing, any large, widely used OSS project has a secondary market for producing plug-ins that extend capabilities—many of which can use proprietary code, depending on the license; distributions or "distros" that package OSS in ready-to-use, user-friendly formats; and providing technical support and maintenance to organizational users of OSS.

Despite these many benefits, there are some effects of the OSS movement that do not always work to an organization's advantage [3, 4]. One of the primary arguments against OSSD continues to be that contributors risk releasing sources of potential competitive advantage. While organizations contribute to OSSD because they hope to obtain the contributions of others, there are cases where a sponsor organization's contributions might be disproportionately large compared to those of others. Although such contributing organizations might hope to obtain intangible benefits such as goodwill and future cooperation, such over-contribution might not always be to their best economic interest. It is wise to carefully evaluate the potential return on investment before making contributions that can prove costly—both in terms of cash expenditures and lost opportunity costs.

Effects on users of open source software

One of the most striking effects of OSSD is that the classical distinction between software developers and end users has been significantly blurred [6]. Under the OSS umbrella, developers and users belong to the same entity—as seen in the principle of most projects being a developer's personal itch-scratching, the developer more often than not *is* the user. Furthermore, other than developer-users, OSSD explicitly emphasizes high user input and participation in the

feature specification process. Users actively participate in beta testing of software—and the OSS philosophy of continual evolution means that most projects are perpetually in beta, thus giving users abundant opportunity to participate. In addition to finding bugs, users may suggest modifications, improvements, and future functionalities. The OSS community generally places a high priority on listening to users—if a project is defective in this area, it is quickly "forked"; that is, the open license is used to take the software and start a separate project that is more responsive to user needs. Such forked projects, especially when the original is insensitive to users, sometimes eclipse their original sources in quality and usage, such as the Firefox web browser, forked from Netscape Navigator.

One negative concern in using OSS is that questions of legal responsibility are sometimes uncertain. Because OSS projects are usually quite open to contributions from anyone, as long as they are actually useful, it is hard to keep ignorant or unscrupulous people from contributing copyrighted code into the project, thus legally jeopardizing the contributions of others. In fact, the SCO Group sued various Linux users for alleged copyright violation of UNIX code, though most analysts considered their claims baseless ["SCO-Linux controversies", 50], and their claims were not held up in court. Nonetheless, Microsoft (who produces the competing Windows operating system) tried to capitalize on users' fears of intellectual property violations to persuade users to avoid Linux and other open source solutions [45].

In addition to these general effects, in this section we discuss effects of the OSS movement specific to consumers and organizational users of software.

Consumers

For consumers, OSS provides an amazing array of inexpensive or free software that is often of very high quality. While some high quality proprietary freeware has always existed,

such applications have usually been small and relatively inconsequential. However, OSS includes incredibly complex user applications of high quality such as operating systems (most notably Linux), virtual machines (Innotek VirtualBox), office application suites (OpenOffice, Gnome Office, KOffice), Internet clients (the Mozilla suite), and many others. OSS gives consumers a wide array of real quality choices beyond traditional proprietary applications. Moreover, not only are the OSS applications free, but proprietary competitors are often forced to drop their prices in order to compete (such as with the Opera web browser, which became freeware to compete with Firefox)—thus OSS benefits even consumers who continue to pay for or use proprietary solutions.

Concerning technical support, on one hand, developers of OSS products typically provide very little, if any, end-user support. On the other hand, online support forums are typically abundant with users and co-developers who are very willing to offer help for free, in the spirit of open source sharing. While this more than adequately meets the needs of more technically inclined consumers, the absence of traditional support such as good help files and telephone support does present a challenge to the common consumer.

Organizational

Insomuch as enterprise-class software is more expensive and is required in greater numbers than for individual consumers, the savings are accordingly greater for organizational users. Moreover, OSS, when free, negates the troublesome problem of tracking the legality of software licenses in use.

Other than cost savings, one major selling point for OSS is that it prevents exclusive dependency on a single developer [35]. A major strategy used by large software vendors is to maintain high switching costs that make it difficult to switch vendors, implemented by various

means such as using proprietary data formats, and maintaining tight access to the source code. However, OSS users are not locked in with the producer of the software. If a producer ceases to support a product for any reason, or the customer is dissatisfied with certain features or lack thereof, access to the source code permits them to hire programmers to make necessary updates and customizations. Thus, it becomes less risky to acquire OSS from lesser-established software producers.

Concerning security, OSS allays some organizations' fears of malicious code: by providing full access to the source code, concerned organizations can be assured that there is no malicious code being acquired. Moreover, with open access, they can rapidly fix any security holes found. This is a particular concern for some governments outside the United States, the country that produces the bulk of the world's software. In order to avoid dependence on foreign (e.g. American) software [7], some governments such as China's are implementing concerted attempts to develop their major software applications on open source platforms such as the Linux operating system (as opposed to the world-dominant Microsoft Windows) [23].

In spite of the benefits, there is one point of caution for organizations implementing OSS. The low price tag might offer a false economy in cases where the quality is inadequate for the organization's needs. The organization might need to factor in employee training costs, as well as product support and customization costs. In contrast, vendor support costs are usually built into the price of most commercial proprietary offerings. Governments might be particularly susceptible to this pitfall, as it is more likely for them to mandate the use of open source software for philosophical reasons rather than based on sound business principles.

Effects on societies as a whole

OSS has some potential to make significant changes in a societies' consideration of

software and other intellectual property. In more economically advanced societies, OSS could help educate and even inculcate the society in the philosophy of intellectual property rights. Moreover, the more OSS is used, the more its gift culture might diffuse into society [2], though some criticize this "anti-capitalistic" tendency as having a debilitating effect on economic incentives for creating certain software products.

Perhaps the most significant societal impacts of OSS might be its potential effects on the information societies of developing nations. While the most apparent benefit might be the availability of software free of charge, in fact, the high incidence of software piracy assures that users are not necessarily deprived of high-quality, expensive software—unavailability of computing hardware is a far more limiting factor in developing countries. Far more significant is the access that OSS provides to the source code, thus giving developers control over the software they use.

Especially in Africa, former European colonial powers have been severely criticized for the economic structures of dependency they left in place when they granted political independence to their former colonies. The economic dependency in the area of software development might not quite be a vestige of colonialism, but it could be a manifestation of continued control of richer nations over poorer ones [39]. Similar to its effect on junior individual developers, mentioned earlier, OSS gives developers in developing nations access to the intellectual capital of developed nations. The vast majority of the world's software is created by developed nations. The knowledge of creating such software is developed and transmitted in their school systems, and is nurtured by their vibrant software industries. Unfortunately, most developing nations—though by no means all, India [26, 34] and China being notable exceptions—find themselves out of the loop, and their programming students have little access to

the world-class code that their countries import from richer nations. OSS gives them direct access to such high-quality code, and permits them to have real professional products that they can learn on and contribute to, thus enhancing the skill pools of their local software industries.

In addition to giving developers in developing nations a genuine opportunity to participate in the international software development community, OSS resolves another major problem in breaking organizational users' dependency on consultants and software support technicians from developed nations. It is quite common for organizations in developing nations to fly in teams of consultants to install their software, and even to maintain it when necessary, all adding astronomically to the cost of major software installations [39]. By giving the organizations access to the code, and by empowering local developers to participate professionally in the development process, OSS helps developing nations to shake off such dependencies and make definite advances towards developing vibrant local software industries.

CONCLUSION

The emergence of open source software has radically changed the technological landscape of the computing industry, affecting the strategic dynamics involved in various commercial enterprises, including the interactions among software users, proprietary software developers, hardware manufacturers, and makers of network products alike. This radical software development methodology has not only created extremely high quality software, but has also created a new genre of virtual community with its particular sociological characteristics. Moreover, it is having far-reaching impacts in the way society interacts with software. While most of the effects are positive, giving software users and developers increased choice at zero or low cost, there are some effects worthy of caution, which we have noted. Overall, we believe that the open source movement is significantly shifting the conception of how intellectual property

rights are used to promote software innovation. Particularly now, as the movement matures, shifting from mere ideology to more pragmatic and economic motivations, its communities of software developers and users forge in the open source intersection of the Internet and copyright law with effective solutions to meet shared public software needs.

REFERENCES

1. Baldi, S.; Heier, H.; and Mehler-Bicher, A. Technical opinion: Open courseware and open source software. *Communications of the ACM*, 46, 9, (2003), 105-107.

2. Bergquist, M., and Ljungberg, J. The power of gifts: organizing social relationships in open source communities. *Information Systems Journal*, 11, 4, (2001), 305-320.

3. Bezroukov, N. Open Source Software as a special type of academic research (critique of vulgar Raymondianism). *First Monday*, 4, (1999),

4. Bezroukov, N. A second look at the cathedral and the bazaar. *First Monday*, 4, (1999),

5. Brookshire, R.G.; Yin, L.R.; Hunt, C.S.; and Crews, T.B. An end-user information systems curriculum for the 21st century. *The Journal of Computer Information Systems*, 47, 3, (2007), 81.

6. Carillo, K.D., and Okoli, C. Open Source Software Communities, in Subhasish Dasgupta, ed., *Encyclopedia of Virtual Communities and Technologies*, George Washington University, USA: Idea Group Reference, 2005.

7. Carmel, E. American software hegemony in packaged software and 'The culture software'. *The Information Society*, 13, 1, (1997), 125-130.

8. Chengalur-Smith, S., and Sidorova, A. Survival of Open-Source Projects: A Population Ecology Perspective, in *Proceedings of* Seattle, Washington, 2003.

9. Dempsey, B.J.; Weiss, D.; Jones, P.; and Greenberg, J. Who is an open source software developer? *Communications of the ACM*, 45, 2, (2002),

10. Diker, V.G., and Scholl, H.J. The art of leveraging: how powerful nonlinear feedback processes can restructure rapidly growing technology and knowledge industries, in *Proceedings of HICSS 2001*, Island of Maui, Hawaii, 2001. IEEE Computer Society, p. 9.

11. Edwards, J. The changing face of freeware. *IEEE Computer*, 31, 10, (1998), 11 - 13.

12. El-Shinnawy, M., and Vinze, A.S. Polarization and persuasive argumentation: A study of

decision making in group settings. *MIS Quarterly*, 22, 2, (1998), 165.

13. Feller, J., and Fitzgerald, B. A framework analysis of the open source software development paradigm, in *Proceedings of 21st International Conference on Information Systems*, Brisbane, Australia, 2000. Association for Information Systems, pp. 58-69.

14. Fitzgerald, B. A critical look at open source. *IEEE Computer*, 37, 7, (2004), 92 - 94.

15. Gacek, C., and Arief, B. The many meanings of open source. *IEEE Software*, 21, 1, (2004), 34 - 40.

16. Ghosh, R.A.; Glott, R.; Krieger, B.; and Robles, G. *Free/Libre and Open Source Software: Survey and Study*, Report of the FLOSS Workshop on Advancing the Research Agenda on Free/Open Source Software, European Commission, (2002).

17. Glass, R.L. Practical programmer: A sociopolitical look at open source. *Communications of the ACM*, 46, 11, (2003),

18. Hann, I.-H.; Roberts, J.; Slaughter, S.A.; and Fielding, R. Economic Incentives for Participating in Open Source Software Projects, in *Proceedings of* Barcelona, Spain, 2002.

19. Hars, A., and Ou, S. Working for free? Motivations of participating in open source projects, in *Proceedings of* Island of Maui, Hawaii, 2001. IEEE Computer Society, p. 9.

20. Holmstrom, B. Managerial Incentive Problems: A Dynamic Perspective. *Review of Economic Studies*, 66, (1999), 169-182.

21. Howard, R. *The Virtual Community: Homesteading on the Electronic Frontier*. Mass: Addison Wesley, 1993.

22. Krogh, G.v.; Spaeth, S.; and Lakhani, K.R. Community, joining, and specialization in open source software innovation: a case study. *Research Policy*, 32, 7, (2003), 1217-1241.

23. Kruck, S.E., and Yu, P. Confronting the OS rivalry: UNIX vs. NT. *The Journal of Computer Information Systems*, 41, 1, (2000), 94.

24. Lakhani, K., and von Hippel, E. How open source software works: "Free" user-to-user assistance. *Research Policy*, 32, 6, (2003), 923-943.

25. Lakhani, K.; Wolf, B.; Bates, J.; and DiBona, C. *The Boston Consulting Group Hacker Survey*, Boston Consulting Group and Open Source Developers Network, (2002).

26. Lal, K. Institutional environment and the development of information and communication technology in India. *The Information Society*, 17, 2, (2001), 105-118.

27. Lerner, J., and Tirole, J. Some simple economics of open source. *Journal of Industrial*

Economics, 50, 2, (2002), 197-234.

28. Ljungberg, J. Open source movements as a model for organising. *European Journal of Information Systems*, 9, 4, (2000), 208-216.

29. Lomerson, W.L.; Jones, C.G.; and Schwager, P.H. Core web technologies for new e-commerce employees. *The Journal of Computer Information Systems*, 45, 2, (2004), 44.

30. Mishra, B.; Prasad, A.; and Raghunathan, S. Quality and Profits Under Open Source Versus Closed Source, in *Proceedings of* Barcelona, Spain, 2002.

31. O'Reilly, T. Lessons from open-source software development. *Communications of the ACM*, 42, 4, (1999), 32-37.

32. OSI, The Open Source Definition. 1997, Open Source Initiative: Internet.

33. Ousterhout, J. Free software needs profit. *Communications of the ACM*, 42, 4, (1999),

34. Rao, S.S. Information systems in Indian rural communities. *The Journal of Computer Information Systems*, 44, 1, (2003), 48.

35. Raymond, E.S. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. Sebstapol, CA: O'Reilly, 2001.

36. Raymond, E.S., Keeping an Open Mind. 2003,

37. Raymond, E.S. Up from alchemy [open source development]. *IEEE Software*, 21, 1, (2004), 88-90.

38. Sharma, S.; Sugumaran, V.; and Rajagopalan, B. A framework for creating hybrid-open source software communities. *Information Systems Journal*, 12, 1, (2002), 7-25.

39. Sohne, G., Discussion on e-business in Ghana, August.

40. Stamelos, I.; Angelis, L.; Oikonomou, A.; and Bleris, G.L. Code quality analysis in open source software development. *Information Systems Journal*, 12, 1, (2002), 43-60.

41. Stewart, K.J., and Ammeter, T. An Exploratory Study of Factors Influencing the Level of Vitality and Popularity of Open Source Projects, in *Proceedings of* Barcelona, Spain, 2002.

42. Stewart, K.J., and Gosain, S. An Exploratory Study of Ideology and Trust in Open Source Development Groups, in *Proceedings of* New Orleans, Louisiana, 2001.

43. Thomas, D., and Hunt, A. Open source ecosystems. *IEEE Software*, 21, 4, (2004), 89 - 91.

44. Uchida, S.; Monden, A.; Ohsugi, N.; and Kamiya, T. Software analysis by code clones in

open source software. *The Journal of Computer Information Systems*, 45, 3, (2005), 1.

45. Vaughan-Nichols, S.J., Get the Facts Right on Linux, Microsoft, in eWeek. 2005,

46. von Hippel, E., and von Krogh, G. Innovation by user communities: Learning from open-source software. *MIT Sloan Management Review*, 42, 4, (2001), 82-86.

47. von Hippel, E., and von Krogh, G. Open source software and the "private-collective" innovation model: Issues for organization science. *Organization Science*, 14, 2, (2003), 209-223.

48. von Krogh, G. Open-source software development. *MIT Sloan Management Review*, 44, 3, (2003), 14.

49. West, J., and Dedrick, J. Proprietary vs. open standards in the network era: an examination of the Linux phenomenon, in *Proceedings of* Island of Maui, Hawaii, 2001. IEEE Computer Society, p. 10.

50. Wikipedia, Wikipedia: The free encyclopedia. 2006, Wikimedia Foundation:

51. ZDNet News, IBM to invest $100 million in Linux push. 2005,

52. Zhao, J.J. Computer end-user skills important for business professionals now and toward 2005. *The Journal of Computer Information Systems*, 42, 3, (2002), 31.