e-mentor
No 5 (27) / 2008
www.e-mentor.edu.pl/eng

John R. Bork

The Free, Open Source Option as Ethic

A new pattern of technical decision making that has had a profound effect on the software industry is to consider free, open source options, wherever feasible, alongside traditional build versus buy alternative when designing and implementing any type of technological system. “Free, open source” (FOS) refers to the permission granted to users to freely use, copy and share technical products, and in the case of computer software, to examine, modify, and redistribute the source code from which the software is generated. The exact terms depend on the license with which the software and its source code are distributed, the most common being the GNU General Public License (GPL). Richard Stallman and the Free Software Foundation articulate these concepts including the definition of FOS and the term copyleft. The FOS option can be articulated as an ethical approach since it guides action and has practical, moral, and epistemological implications for being used instead of its not free, not open source counterparts. I leverage Carl Mitcham's approach to philosophy of technology studies to elucidate the practical and moral ethical aspects along the lines of the engineering and humanities perspectives. For practical engineering, I stress the importance of presenting correct, up to date information about free, open source software in computer ethics. This includes attention to technical details garnered from the technologist's perspective, and the utilization of empirical studies to counter misconceptions spread by both evangelists and detractors. For humanities the FOS option raises moral issues over the value of freedom itself, the obligations of government to provide unhindered access to public services, and the interest of sovereign states to control their national information. Additionally, I explore its relationship to Andrew Feenberg's theory of technological transformation and address the problem of the alienation of intellectual labor. Finally, I suggest that there are largely unexplored epistemological facets to the topic that arise from the synthesis of the engineering and humanities approaches. This paper is based on a presentation of the same title made at the 2007 meeting of the International Association for Computing and Philosophy at Loyola University, Chicago[1].

Introduction

The conceptual muddles and policy vacuums attending the absence of a formal philosophy of computing technology have been filling with ideas emanating from technical fields in addition to affiliated academic disciplines. These include the Association for Computing Machinery (ACM), the Institute of Electrical and Electronics Engineers (IEEE), and most notably the International Association for Computing and Philosophy (IACAP) and the American Philosophical Association's committee on Philosophy and Computers. An important example is the emergence of the free, open source (FOS) movement popularized by global, inter networked software development communities and users. A decade ago words like 'Linux' and 'open source' were unknown to technology managers and never came up in feasibility studies, although behind the scenes integrators were creating solutions leveraging technologies like GNU, Linux, Apache, and MySQL, and were spending work hours improving source code made freely available on global developer communities like Sourceforge.net. A presentation titled *Linux: It's Philosophical Significance* by Herbert Hrachovec at a 2000 meeting of IACAP drew little notice. It was seldom the subject of computer ethics, either. There is nothing in the third and still current 2001 edition of Deborah Johnson's *Computer Ethics* besides a footnote where she incorrectly characterizes “the Linux operating system” as “shareware” (Johnson, 2001, p.160). Perhaps because she is a software consumer rather than a software developer, her perspective on ethical questions focuses on whether to pay for her software or engage in piracy, and how software owners can protect their intellectual property. She does not question whether the model of software development based treating it as property to be protected from others is optimal. But as Richard Stallman famously points out, what we are interested in is 'free' as in “free speech,” not “free beer” (Stallman, 2002c). Now that free, open source software is firmly entrenched, it is getting serious attention from technology scholars, as we find, for example, in the 2005 volume *Perspectives on Free and Open Source Software*. It is incumbent upon philosophers studying computing and technology to join the debates. Johnson's footnote should be replaced by an entire chapter on its ethical implications. This study will outline three different approaches on how to do so.

What is the FOS Option?

[1] The current article is a version of a paper at the 2007 North American Computing and Philosophy conference at Loyola University, Chicago.

Free, open source software (FOSS) is often defined by the General Public License (GPL) developed by the Free Software Foundation. Its four main components are: (Freedom 0) freedom to run a program for any purpose; (Freedom 1) freedom to modify the program source code; (Freedom 2) freedom to redistribute copies of the original or modified program; and (Freedom 3) freedom to redistribute the modified source code (Stallman, 2002a). It is understood that access to the source code itself is a precondition to all four of these freedoms. The Open Source Definition version 1.9 from the Open Source Initiative is another common reference for clarifying the idea of FOSS, in this case by ten attributes: permit free redistribution, include source code, allow derived works, preserve integrity of author's source code, not discriminate against persons or groups, nto discriminate against fields of endeavor, distribute the license, not tie license to a product, not restrict other software, and be technology neutral (OSI, 2006). Technology evangelists like Richard Stallman, Eric S. Raymond, and Tim O'Reilly have promoted this idea into an ethic to live by for technology professionals and general users alike. In recent years, countless individuals and organizations have been replacing internally developed and commercial applications with FOSS, and directing software development activities towards FOS communities. The idea of "open source" has grown in popularity to the extent that it is used outside of computer software to describe a way of doing activities as diverse as biological science and brewing beer. The reasons range from practical, economic benefits, to moral imperatives, to enhancements to learning, and to the sheer joy of creativity. These motivations are gathered under the rubric of the "free, open source option". My goal is to argue in favor of the free, open source option as an ethic to guide decision making based on its practical, moral, and epistemological advantages over conventional proprietary, non-free options.

Richard Stallman founded the GNU's not Unix (GNU) project and the Free Software Foundation (FSF) with the aim of helping researchers, educators and hobbyists to legally possess, modify, and redistribute the source code for the components of a UNIX-like operating system. He did so in response to the loss of this freedom when DEC made the original Unix source code proprietary in the name of corporate self-interest in the 1980s. He gathered the four kinds of freedom previously noted into the legal concept "copyleft", formalized in the GPL, as a play on the term copyright. It is important to differentiate this powerful legal concept from shareware and freeware. For instance, Deborah Johnson equivocates them in a footnote of the chapter in *Computer Ethics* on "Property Rights in Computer Software" when she states that "the best example of successful shareware is the Linux operating system" (2001, p. 160). Neither the Linux kernel nor the GNU components of such operating systems are shareware (and technologists will point out that Linux is not an operating system at all, only the kernel). Shareware permits the redistribution of copies, but beyond a fixed number of days users are required to a pay a licensing fee; its source code is seldom made available (Free Software Foundation, 2007). Freeware, which does not require payment to use, typically does not make available or allow modification of the source code (Free Software Foundation, 2007). There are also licenses that permit read-only access to source code, such as the upcoming Microsoft Reference License, but none of the four freedoms granted by the GPL (Hoover, 2007). Free, open source software, on the contrary, when defined by a license such as the GPL, ensures the four freedoms listed above are retained. The free, open source option as ethic can be concisely stated as: "Consider free, open source options, characterized by licenses such as the GPL, wherever feasible."

Why Consider the FOS as an Ethic?

James Moor suggested that "conceptual muddles" and "policy vacuums" exist where there are problems lacking a philosophical framework to address them, and this is particularly true of computer technology (Moor, 1985). Likewise, Walter Maner proposed that innovations in computer technology create unique, new ethical problems (Maner, 1995). For years, this conceptual vacuum has been filling with the musings of self-proclaimed accidental revolutionaries like Richard Stallman, Eric Raymond, and Linus Torvalds, the creator of the Linux kernel, as well as industry leaders like Bill Gates and Tim O'Reilly. While subject area experts have arisen in the field of computer ethics and the philosophy of computing and information, articulation of the ethical implications of trends favoring free, open source software are only beginning to be featured in academic publications and conferences. An excellent example is the 2007 North American meeting of IACAP, which keynoted free software and open access. The argumentative approach I have selected is borrowed from the philosophy of technology, in particular the work of Carl Mitcham and Andrew Feenberg, to present practical and moral advantages of the FOS option. Finally, I will offer a third approach based on its potential epistemological advantages.

The Engineering Philosophy of Technology Approach

In *Thinking through Technology: the Path between Engineering and Philosophy*, Carl Mitcham introduced the Engineering Philosophy of Technology (EPT) as the field of study focused on determining the best way to conduct engineering and technological endeavors (Mitcham, 1994). This work is from the insider's perspective, and the obvious starting point to transfer insights from the technical arena to the academic study of FOSS. There is a ready set of commonly cited practical benefits supported by empirical research as well as the methodologies used to evaluate, organize, and execute such projects (Lerner and Tirole, 2005). Practical ethics have to do with making everyday choices and judging which are appropriate based on their anticipated outcome. In this respect, technologists engage ethics in the early stages of project management

when they evaluate options. A fundamental differentiation of options to be considered has always been between in-house *versus* third party, or build *versus* buy (Weinstock and Hissam, 2005). Other 'practical ethics' employed by technology decision makers include minimizing the total cost of ownership (TCO), using the best tool for the job, standardizing on a particular technology tool set, and outsourcing where there is no competitive advantage, which is to leave the decision to a third party. One ought to add, "utilizing free, open source options where feasible."

The free, open source option presents a significant divergence from the traditional ethical approaches that focus on how much third party software costs. Copyleft licensing affects the way technology workers interact with the product in all stages of its life cycle, potentially fostering synergies between the advantages of having in-house personnel with direct access to source code and having a viable external community to sustain it (Lerner and Tirole, 2005). The benefits of custom, in house development are therefore joined with the benefits of outsourcing the majority of support functions, which is particularly useful for "back office" technical operations that do not yield a competitive advantage. Eric Raymond, self proclaimed cultural anthropologist and accidental revolutionary, suggests in *The Cathedral and the Bazaar* that the synergies resulting from an environment characterized by greatly diminished transactional costs, many eyes examining shared code − signified by the decentralized, unregulated bazaar −
can out compete traditional, closed, tightly regulated hierarchical modes of software development, signified by the cathedral (Raymond, 1999). Raymond theorizes that FOSS communities operate as gift cultures, and hackers are motivated to participate for status gains related to their ability to contribute intellectual labor. Pekka Himanen's *The Hacker Ethic* delves deeper into the psychological motivations behind participation in FOS projects, as does Linus Torvalds' autobiographical *Just for Fun*. A primary task for EPT is to help distinguish these anecdotal analyses of the FOS phenomenon from the facts suggested by empirical research. The recently published collection *Perspectives on Free and Open Source Software* provides analysis of thorough, empirical studies on its motivation, economic benefits, and social dynamics. This includes dismantling the myth that participation in FOS projects is purely grounded in altruism. A large scale survey of official developers listed in Sourceforge.net projects disclosed an abundance of selfish motives and the fact that large contributions to many FOSS projects occur in the context of paid employment (Lakhani and Wolf, 2005). The greatest factor is the sense of creativity. The review the European FLOSS study by Rishab Aiyer Ghosh suggests "balance value flow" accounts for non-monetary rewards to sustain underlying rational self-interest motive over altruism (Ghosh, 2005).

A key practical decision area is determining where the FOS option is appropriate, despite the desire of evangelists like Stallman to deploy it everywhere. Projects need a critical mass of developers in addition to a user community, and the best projects "scratch an itch" (Weinstock and Hissam, 2005). Furthermore, a reasonable tool base must be adopted including revision control and bug tracking, as in the very popular Sourceforge.net. A recurring theme is that FOSS selection must be aligned with the capabilities of the enterprise. Open source projects often lack the assurances and backing of commercial, off the shelf software (COTS), and without knowledgeable in-house staff, having the source is pointless (Fitzgerald, 2005). The open source model may prevail over proprietary models because the latter tend to rush to market and fail to focus on security and reliability. The profit-driven model therefore can limit which projects are permitted to mature (Neumann, 2005; Anderson, 2005).

To be fair, FOS is not without its critics. The popular press is quick to publicize comments from proprietary software vendors such as Microsoft when they accuse FOSS of spreading communism and attacking American business models based on licensing intellectual property. This is often referred to as Fear, Uncertainty, and Doubt (FUD). While it may be true, as Selmer Bringsjord comments in The *Irrationality of the Free Software Movement*, that ultimately lawyers will decide the fate of FOSS, this task also devolves to the practitioners of EPT. It is clear that successful businesses can be created around copylefted source code, but are these business models sustainable? Brian Fitzgerald presents a list of problematic issues from software engineering, business and sociocultural perspectives that goes far in replacing FUD with genuine issues (Fitzgerald, 2005). The widely cited *Economic Perspectives on Open Source* by Josh Lerner and Jean Tirole summarizes four open economic questions about open source: which technological characteristics are conducive to smooth open source development, what is the optimal licensing arrangement, how can commercial and open source software coexist, and whether the process can be transposed to other industries (Lerner and Tirole, 2005). Even the sacrosanct presumption that open source leads to better quality code because there are many eyes looking at it has been attacked by a careful study of the Address Resolution Protocol (ARP) module in the Linux kernel, which revealed a very convoluted, poorly documented design (Rusovan, Lawford, and Parnas, 2005). EPT practitioners can help provide reasoned argument grounded in empirical research to replace propaganda.

The Humanities Philosophy of Technology Approach

Mitcham's second approach to thinking through technology is from the perspective of its moral, psychological, social, and environmental consequences. This perspective, coming from the humanities, is often referred to as the outsider's perspective, which he calls the Humanities Philosophy of Technology (HPT) approach. Here the accidental philosophers of the free, open source movement currently have more to say than their professional counterparts in academia. Johnson's aforementioned treatment of open source in *Computer Ethics* reflects the limited scope of reflection on free software when

it is grounded in terms of traditional intellectual property debates. Some of the often-cited moral considerations advantaging the FOS option include examining the value of freedom in itself, the problems with universalizing intellectual property claims, and new ethical problems related to public services. By extending the work of the HPT theorist Andrew Feenberg, they also include transforming technology and addressing the problem of the alienation of intellectual labor.

Software piracy is very tempting due to the relatively high cost of commercial applications, the easy transfer of digital information, and the lack of a perception of doing harm. Software piracy is especially common among curious academics and hobbyists; why not avoid the moral dilemma by selecting FOSS? Stallman insists that the value of freedom in itself must be considered in addition to pragmatic arguments. He takes aim at the use of the term 'pirate' to refer to those who copy and distributed non-free software, suggesting that in the interest of profiting from intellectual property, sharing software has been demonized, equated with violently robbing people and attacking ships (Stallman, 2002d). Public policy has been influenced by donations from large corporations so that children are taught that it is wrong to help your neighbor. He goes as far as to invoke Kantian ethics to argue that use restrictions, including charging a fee, reduce the wealth humanity as a whole derives from software. If everyone used this destructive means to become wealthier, "we would all become poorer from the mutual destructiveness" (Stallman, 2002b). He uses a similar universalizing argument when criticizing software patents. As the body of patents grows, only large corporations who cross-license each other's intellectual property will be able to innovate (Stallman, 2002). Thus, using an entirely FOSS platform allows one to take to the moral high ground. It also provides guidance as to what actions are permissible with respect to licensed material. The DMCA prohibits reverse engineering copy protection schemes, which in turn prevent unlicensed redistribution of machine code. Copyleft clears ambiguity so that researchers can avoid such traps (Lessig, 2005; McGowan, 2005).

From the perspective that the FOS movement is something new, additional moral arguments can be based on Walter Maner's notion that electronic computing technology creates unique ethical problems. For example, there evolves the question of whether there should be unhindered access to public information and services that are compatible with FOS technologies whose use have become widespread, as well as allowance for FOS options to compete in publicly funded initiatives. Why do we, Americans, have to pay for proprietary tax preparation software just to obey the law, or use Microsoft products to access websites providing government services? Controversy rages over the adoption of open document standards by state governments; lobbyists from organizations including the Electronic Frontier Foundation and the Free Software Foundation compete with those from Microsoft in the effort to influence policy decisions. A good example is the work of the President's Information Technology Advisory Committee (PITAC), who found there were no guidelines for authorizing competition between open source and proprietary software. It has since made recommendations to remove barriers and educate decision makers (Weinstock and Hissam, 2005). There may be an obligation to untangle national sovereignty from reliance upon foreign held proprietary technology products and services. The ethical question also arises whether low-cost computers should be distributed to help disadvantaged social groups; technological solutions like the One Laptop Per Child initiative highly leverage FOSS.

The free, open source option fits nicely with Andrew Feenberg's theory for advancing social change by transforming technology. Indeed, the revised version of *Transforming Technology* published in 2002 has a new section on "the Ambivalence of the Computer." His main argument is that the long history of bureaucratic control in large corporations has crystallized into self-serving organizational structures that perpetuate and strengthen capitalism itself. It is this 'reifying' tendency, not something inherent in the nature of modern technology itself as other theorists have argued, that has resulted in the current state of affairs in which human progress seems constrained by the short-sighted goals of transnational business (Feenberg, 2002). Technology can be transformed by decoupling it from these processes and extending the scope of planning to encompass goals commonly associated with socialist societies. The computer is ambivalent because it can be used to further enforce control or foster flexibility. He looks to the example of 'rationalizations' within the former Soviet union. Free, open source software projects share an uncanny resemblance:

> *Workers were offered a means of claiming authorship and receiving bonuses for useful ideas. To promote worker participation in innovation, 'complex brigades' of workers, engineers, and others were assembled to draft blueprints, test solutions, and refine original ideas. Several mass organizations mobilized large voluntary support networks to help worker-innovators overcome the bureaucratic obstacles to success. ... Capitalist management and product design aims to limit and channel the little initiative that remains to workers and consumers. Their margin of maneuver is reduced to occasional tactical gestures. But the enlargement of margin of maneuver in a socialist trajectory of development would lead to voluntary cooperation in the coordination of effort.* (Feenberg, 2002, p.157 and p.183).

Regarding margin of maneuver, the FOS option offers a partial solution to the problem of the alienation of intellectual labor, too, an insight gleaned from my personal experience as a professional software engineer and hinted at in many of the aforementioned FOSS studies. Most software developers sign away the right to review, let alone utilize, the code they write, in non-disclosure and non-compete agreements that are a condition of employment. This not only alienates them from the product of their labor, but prevents the "standing on the shoulders of giants" that sustains the successful transfer of knowledge in the sciences (Torvalds and Diamond, 2001). By allowing software professionals to contribute to FOS projects that are used by their enterprise, a portion of their intellectual labor can escape the black hole of non-disclosure agreements.

Not only can past work be reused and referenced, but it can be used by future employers to evaluate a candidate's competency. Could any scholar imagine being deprived of the right to reference his or her past work after leaving a particular institution? Such concerns have a Socratic ring to them, and lead to the final approach to examining the free, open source option as ethic.

The Epistemological Approach

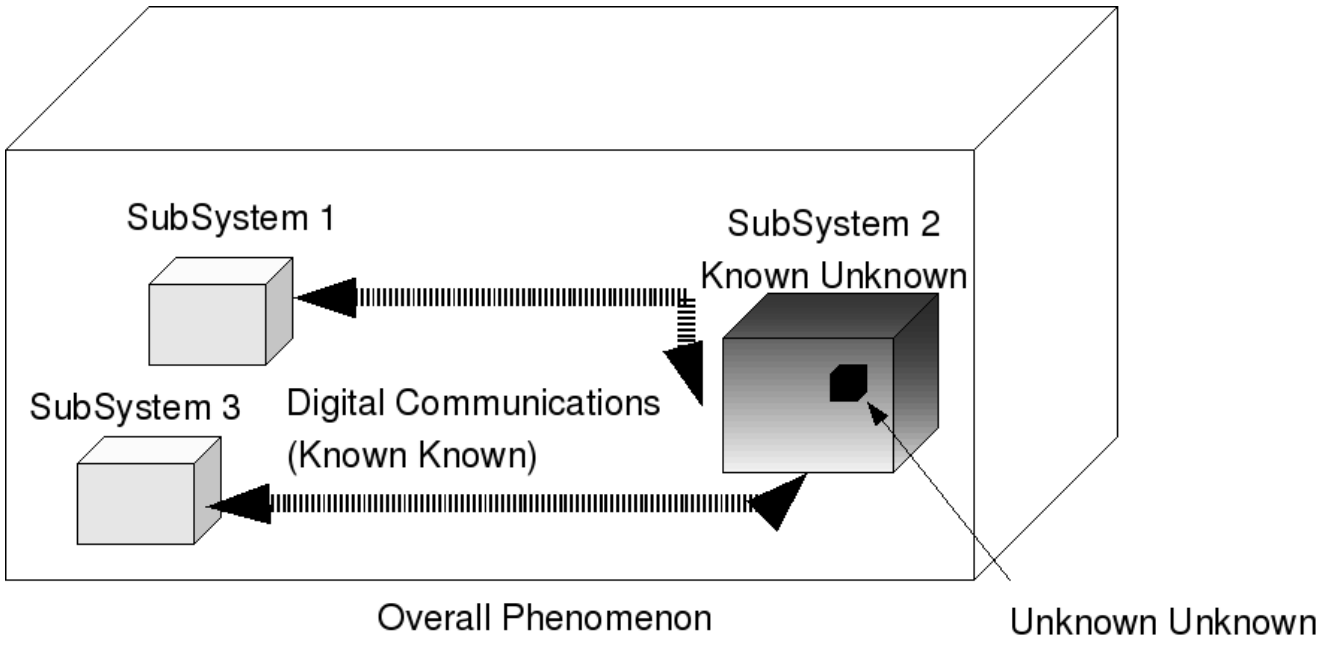
The third way to think about the FOS option as ethic focuses on its epistemological implications. Consider the following quote from Donald Rumsfeld, former US Secretary of Defense:

> *Reports that say that something hasn't happened are always interesting to me, because as we know there are known knowns; there are things we know we know. We also know there are known unknowns; that is to say we know there are some things we do not know. But there are also unknown unknowns--the ones we don't know we don't know.* (Rumsfeld, 2002)

Nowhere do we feel we have the potential to know everything, for being completely derived from design specifications and behaving deterministically, while at the same time to come across known unknowns − black boxes − that may contain further unknown unknowns, than when dealing with complex, stored program electronic computer systems. Technological artifacts have a distinct ontic level "characterized by properties and laws of its own" (Bunge, 1979). Moreover, many technological systems have evolved haphazardly and are therefore very difficult to understand (Winner, 1977). Compound those features with the obfuscated, costly 'black box' exegesis required to reverse engineer closed source systems and even more time and money is wasted when it comes to trying to understand how technologies work. Thus the search for understanding can be thwarted by structural, accidental, and legal constraints − or be epistemologically transparent. Free, open source defaults to a 'white box' understanding of phenomena under its care. A cardinal rule of the GPL is that the source code must be made available, either as part of the product or via convenient means. Thus, taking the free, open source path when learning about technologies such as TCP/IP networking and computer operating systems provides an epistemological advantage over approaches that inevitably and pointlessly suffer detours around deliberate black boxes. It can be extended to the technological topics in general: when studying a particular technology in order to grasp general concepts, choose free, open source options to avoid the unnecessary obfuscation motivated by the desire to hoard intellectual property.

Let us take the example of cyberspace, a term coined by John Perry Barlow for "the present-day nexus of computer and telecommunications networks" (Wikipedia, 2007). Philosophers grapple with its phenomenology and implications, while its technical nature is clearly understood by engineers. Transmission Control Protocol/Internet Protocol (TCP/IP) is the backbone of the Internet, in which every node can be analyzed as either a host or a router, depending on its function (Hunt, 1998). TCP/IP and the other communications protocols that are wrapped inside it, such as Hypertext Transport Protocol (HTTP) and Hypertext Markup Language (HTML), are all defined by open standards known as Request for Comments (RFCs). A router's primary function is to forward packets according to well defined rules; along with the actual transmission media they make up the part of cyberspace that relays information, but does not originate it. Hosts are the sources and consumers of that which is encoded within these protocols and forwarded throughout the network; they are the points at which human beings interface cyberspace, too.

Figure 1. Typical client/server HTTP-based communications system consisting of a webserver (SubSystem 2) and browser hosts (SubSystem1 and SubSystem3) connected via routers.



Viewed from the functional perspective of the browser hosts, one the one hand, in which the communications protocols implemented by the browser software and the operating system are the common, well defined protocols in everyday use, these hosts and the intermediate network are epistemologically transparent. The inner workings of the webserver host, on the other hand, is opaque, not immediately accessible from either client host, and must be evaluated by its responses to the clients' requests. Using Rumsfeld's terms, the client hosts and intermediate network are *known knowns*, to the extent that they correctly implement open standards communications protocols and can be directly examined to contain only FOS components, whereas the browser host is functionally a *known unknown*, unless of course all of its relevant source code and data are made available, or its exact specifications are published. In fact, the webserver may contain *unknown unknowns*, custom applications behind the public interface. These are quite possibly impenetrable black boxes, for example machine code binaries protected by restrictive licenses and cryptographic obfuscation. In reverse engineering jargon this is "mystery goo". Sometimes even the keepers of the webserver itself do not know everything about it, since it may have been developed over a long period of time by various individuals who may or may not have exactly followed the design specifications, to the extent that there were any, and left any accurate documentation. If the webserver source code is licensed as FOSS, the mystery can be unraveled. For the knowledge seeker trying to grasp the phenomenon of cyberspace, as instantiated by TCP/IP inter networked hosts, the FOS option guarantees the possibility of comprehensive understanding.

It is more than a matter of transparency for individual knowledge seekers. The activity of the social networks that form around FOS projects, the so called "many eyes approach", has been promoted as a superior method for distributed development and quality assurance for creating and maintaining software (Raymond, 1999). Its epistemological hypothesis is expressed by the often cited statement by Linus Torvalds, "with a million eyes, all software bugs will vanish" (Torvalds, 2001, p. 226) Thus the popular counterexample to the presumed, superior quality of open source code, the critical evaluation of the ARP module in the Linux kernel mentioned earlier, itself hinges on access to the source code. A rigorous comparison to its counterpart in proprietary operating systems cannot be made at the source code level (Rusovan, Lawford, and Parnas, 2005). Thus, the availability of code for many eyes to see also aids scholarship. The historical study of software can be enhanced not only through the availability of source code itself, but the history of revisions, the bug reports, and the transcripts of project teams and discussion forums. Proprietary software vendors are very reticent to make this kind of information available to researchers, hindering the scholarly study of software and robbing us of our shared intellectual history (Cortada, 2002).

Conclusion

This study seeks to identify significant philosophical implications of the free, open source option as it has emerged in global software development communities. A three part approach inspired by the Carl Mitcham's philosophy of technology has been employed. Each section has touched on some ideas whose elucidation are in no way complete. A number of tasks for the Engineering Philosophy of Technology were identified: clarifying FOS concepts, debunking myths, and providing frameworks for technical decision making. For Humanities Philosophy of Technology, tasks include moral evaluations of both traditional, property-oriented arguments and those developed by the accidental revolutionaries of FOS, investigating

new ethical problems, and examining large scale social implications. Finally, there are largely unexplored epistemological facets to the topic with implications for individual education, the history of software, and collective problem solving skills.

References:

Anderson, Ross. (2005). "Open and Closed Systems are Equivalent (That is, in an Ideal World)" in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Bringsjord, Selmer. (2007). "The Irrationality of the Free Software Movement." Retrieved October 2, 2007 from http://www.rpi.edu/~brings/sb_idiocy_fsm.pdf.

Bunge, Mario. (1979). "Philosophical Inputs and Outputs of Technology" in *Philosophy of Technology: the Technological Condition*, edited by Robert C. Scharf and Val Dusek. Malden, MA: Blackwell Publishing.

Cortada, James W. (2002). "Researching the History of Software from the 1960s." *IEEE anals of the history of computing*, January-March 2002.

Feenberg, Andrew. (2002). *Transforming Technology: a Critical Theory Revisited*. New York: Oxford University Press.

Fitzgerald, Brian. (2005). "Has open source software a future?" in *Perspectives on free and open source software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Free Software Foundation. (2007). "Categories of Free and Non-free Software." Retrieved October 2, 2007 from http://www.gnu.org/philosophy/categories.html.

Ghosh, Rishab Aiyer. (2005). "Understanding Free Software Developers: Findings from the FLOSS Study" in *Perspectives on free and open source software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Himanen, Pekka. (2001). *The Hacker Ethic and the Spirit of the Information Age*. New York: Random House.

Hoover, J. Nicholas. (2007). "Microsoft to Release .Net Source Code." *Information Week*, October 3, 2007. Retrieved on October 3, 2007 from http://www.informationweek.com/news/showArticle.jhtml?articleID=202200793.

Hunt, Craig. (1998). *Tcp/Ip Network Administration* (second edition). Sebastopol, CA: O'Reilly and Associates, Inc.

Johnson, Deborah. (2001). *Computer Ethics* (third edition). Upper Saddle River, NJ: Prentice Hall.

Lakhani, Karim R. and Wolf, Robert G. (2005). "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects" in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Lerner, Josh and Tirole, Jean. (2005). "Economic Perspectives on Open Source" in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Lessig, Lawrence. (2005). "Open Code and Open Societies" in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

McGowan, David. (2005). "Legal Aspects of Free and Open Source Software" in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Maner, Walter. (1995). "Unique Ethical Problems in Information Technology" in *Computer Ethics and Professional Responsibility*, edited by Terrell Ward Bynum and Simon Rogerson. Malden, MA: Blackwell Publishing.

Mitcham, Carl. (1994). *Thinking Through Technology: the Path Between Engineering and Philosophy*. Chicago: The University of Chicago Press.

Moor, James. (1985). "What is Computer Ethics?" in *Computer Ethics and Professional Responsibility*, edited by Terrell Ward Bynum and Simon Rogerson. Malden, MA: Blackwell Publishing.

Neumann, Peter. (2005). “Attaining Robust Open Source Software” in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

O'Reilly, Tim. (2005). “The Open Source Paradigm Shift” in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Open Source Initiative. (2006). “The Open Source Definition.” Retrieved October 2, 2007 from http://www.opensource.org/docs/definition.php.

Raymond, Eric S. (1999). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol, CA: O'Reilly and Associates, Inc.

Rumsfeld, Donald. (2002). “Department of Defense News Briefing, february 12, 2002.” Retrieved October 2, 2007 from http://www.quotationspage.com/quotes/Donald_H._Rumsfeld.

Rusovan, Srdjan, Lawford, Mark, and Parnas, David Lorge. (2005). “Assessing Open Source Software Development” in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Stallman, Richard M. (2002). “The Danger of Software Patents” in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, edited by Joshua Gay. Boston: GNU Press.

-----. (2002). “The Free Software Definition” in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, edited by Joshua Gay. Boston: GNU Press.

-----. (2002). “The Gnu Manifesto” in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, edited by Joshua Gay. Boston: GNU Press.

-----. (2002). “Why 'Free Software' is Better than 'Open Source'” in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, edited by Joshua Gay. Boston: GNU Press.

-----. (2002). “Why Software Should be Free” in *Free Software, Free Society: Selected Essays of Richard M. Stallman*, edited by Joshua Gay. Boston: GNU Press.

Torvalds, Linus and Diamond, David. (2001) *Just for Fun: the Story of an Accidental Revolutionary*. New York: Harper Collins Publishers.

Weinstock, Charles B. and Hissam, Scott A. (2005). “Making Lightning Strike Twice” in *Perspectives on Free and Open Source Software*, edited by Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. Cambridge, MA: The MIT Press.

Wikipedia. (2007). “Cyberspace.” Retrieved October 2, 2007 from http://en.wikipedia.org/wiki/Cyberspace.

Winner, Langdon. (1977). “Luddism as Epistemology” in *Philosophy of Technology: the Technological Condition*, edited by Robert C. Scharf and Val Dusek. Malden, MA: Blackwell Publishing.

John Bork is a software engineer with a background in process control automation and a doctoral student in the Texts and Technology program at the University of Central Florida. His primary research interest is the intersection of computing and philosophy, in particular the expanding influence of free, open source software. The thesis project for his Master in Industrial Technology from Bowling Green State University was featured in the November 2005 issue of Linux Journal, in article entitled "The Pinball Machine Reverse Engineering Kit".