OPEN SOURCES AND THE OPEN SOCIETY

AN ESSAY IN POLITICS AND TECHNOLOGY

Version 1.0.1[*]

Kelly A. Parker
Grand Valley State University
Allendale, Michigan 49401 USA
email: <parkerk@gvsu.edu>
The original of this document is available at
http://agora.phi.gvsu.edu/kap/OSFS/

10 June 2000

> The grand, leading principle, towards which every argument unfolded in these pages directly converges, is the absolute and essential importance of human development in its richest diversity.
>
> —J. S. Mill, "Dedication," *On Liberty*

Introduction

"Open source" and "free software" (OS/FS) licenses use copyright law to establish key rights for developers and end users. In the language of the GNU General Public License, the aim of so-called "copylefting" is to ensure "that you have the freedom to distribute copies of free software. . . , that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things" (1991, "Preamble").[1] The

[*] Version 0.9 of this paper was presented at a Directions and Implications of Advanced Computing Symposium on "Shaping the Network Society," sponsored by Computer Professionals for Social Responsibility at the University of Washington, Seattle, 20 May 2000. Version 1.0 included revisions; version 1.0.1 introduced only pdf file formatting changes.

[1] A number of other open source licenses are in use, but the GPL is the earliest and is definitive of both "open source" and "free software." It is the most liberal licensing strategy available, aside from simply placing a program in the public domain. See section 2.5, following, for discussion of some other open source licensing strategies.

concerned OS/FS user can understand and control every aspect of the machine; the concerned "closed" or "proprietary" program user can only guess at the inner workings, customize some settings, and hope everything runs as advertised and does what is needed.

The Orbiten Free Software Survey, published in May 2000, identified 3,149 separate programming projects released under such licenses in recent years. The survey cataloged over one gigabyte of code written by 12,706 individuals, all available for unrestricted use, modification, and distribution–in most cases with no licensing fee (Ghosh and Prakash 2000).[2] *Whatever* anyone wants to do with a computer can probably be done very well with some combination of these programs. If not, just call a programmer–the main advantage of open source is that modifications are possible.

C. Scott Ananian observes that we end users "tend to accept our computer's workings as immutable, that we are chained to an irrational, vindictive, uncontrollable machine destined to rule over our 9-to-5 days" (2000). This is an ironic situation, since the singular beauty of a computer is the fact that it can be programmed and is thus in principle highly mutable and entirely subject to user control. Users have had to *learn* to think otherwise; how they did so is a subject for another discussion. OS/FS is a powerful antidote to the servile mindset Ananian describes.

The recent explosion in OS/FS has attracted considerable attention from Wall Street and elsewhere, but it *might* yet prove to be a passing trend. In spite of what its advocates sometimes say, there is nothing inevitable about the movement's success. All cultural movements are, after all, affected by fate and political will. After observing the OF/FS movement for several years, I have developed a hypothesis that I hope will help us to understand and direct the evolution of information technology: *The OS/FS movement is to software development what open debate and peer review are to philosophical, scientific, and political discourse.*

A brief word about the place of this investigation is in order. Eric Raymond, President of the Open Source Initiative, has urged open source advocates to stay clear of philosophy and politics. He believes the movement will be better served by "sticking to relatively narrow, pragmatic arguments" about the economic and engineering advantages of open source software: its lower cost, greater stability, and faster upgrade cycle, for example (1999, 225-27).[3] I do sympathize with Mr. Raymond. Strong, readily available arguments are preferable to potentially weak, undeveloped ones. The facts are, however, that the OS/FS movement arose in a political context, that it relies upon and promotes a recognizable ideology, and that the movement will continue to have social and political effects. These effects, more-

[2] These numbers are certainly low, since this first Orbiten survey did not attempt to catalog all known free software archives.

[3] Similar concerns have led Raymond to promote the boardroom-friendly "Open Source Definition" in place of the less restrictive GNU "Free Software" license (Perens 1997).

over, may prove to be enormous. The same is true of closed proprietary software development and distribution, of course, which makes it all the more important to scrutinize the ideologies at work in *both* models.

Though some more technically oriented supporters of OS/FS might prefer not to worry about politics, it is best to enter this territory with eyes wide open. Others recognize this as well. Richard Stallman of the Free Software Foundation is well known for his passionate arguments about the value of free software (Stallman 1999b; Stallman 1999c). A handful of legal experts have observed and influenced the movement from early on–a clear necessity, given the centrality of "copyleft" licenses.[4] Studies of the economics of OS/FS have begun to appear (Kaminsky 1999). And in spite of himself, even Eric Raymond has recently weighed in with his thoughts on the role of public policy in promoting OS/FS (Raymond et al. 2000). Now is the time to engage an honest discussion about the political and philosophical nature of OS/FS. We need to ask questions about the values it promotes, or that it ought to promote. Such a discussion, it appears to me, is likely to produce political and ethical arguments that are at least as compelling as the pragmatic arguments advanced by Raymond and many others. As only one voice speaking in a relatively new discussion, I have chosen to concentrate on developing the advocate's case. Others will no doubt present the strong counterarguments I am neglecting here.

An analysis of the social and political significance of OS/FS requires that we understand both the *philosophical antecedents* and the *contemporary social-political implications* of the movement. The roots of this movement can conveniently be traced to 18th and 19th century Enlightenment political ideals of John Locke and John Stuart Mill. The classical liberalism exemplified by Locke and Mill shaped the principles and institutions that make the OS/FS movement possible: open deliberation in democratic governance, open access to education, open reasoning in science, and open competition in business. Not surprisingly, it is in these same areas–governance, education, science, and commerce–that we can find the movement's most immediately visible contemporary implications. Based on my understanding of these connections, I argue that the OS/FS movement promises to increase public participation in the systems that create, control, and deliver fundamentally important social goods. My thesis in turn supports the small but growing movement toward OS/FS use and advocacy in government, education, research, non-profit, and commercial institutions.

1 Historical Antecedents

Historically, the "hacker culture" (or, if you prefer, the "do-it-yourself-computing" culture) that created OS/FS arose from several hundred years' experimentation with

[4] Lawrence Lessig is among the most visible legal experts interested in OS/FS.

the political ideals of autonomy and equality. The OS/FS movement thus embodies and tends to promote the same ideals that drive democracy, education, research, and perhaps less intuitively, free enterprise.

In his *Second Treatise of Government*, John Locke described individuals' "natural" political status as follows:

> a state of perfect freedom to order their actions, and dispose of their possessions and persons, as they think fit, within the bounds of the law of nature, without asking leave, or depending upon the will of any other man.
>
> A state also of equality, wherein all the power and jurisdiction is reciprocal, no one having more than another.... (1690, II. 4.)

John Stuart Mill further developed these principles of autonomy and equality in his treatise *On Liberty*. In Chapter II, "Of the Liberty of Thought and Discussion," Mill argued for unrestricted public discussion of any opinions whatsoever and further explored what is implied by the central liberal ideals. He addressed two key questions concerning restricting access to information. First, he considered why we should permit public debate about opinions that are generally "known" to be false or inferior. Such debate sometimes reveals that they are in fact true, which is the most important reason to encourage debate. More often, though, it confirms the belief that they are inferior. Either way, those involved in the debate gain in terms of basic autonomy, their ability to determine beliefs and actions for themselves. Where there is not full understanding of the range of live alternatives, there can be no genuinely free choice among them. Where there is not full disclosure of the weaknesses of even the best among the alternative beliefs, there cannot be valid consent to actions taken on the basis of those beliefs. Thus open deliberation is essential to autonomy, both at the level of individual choice and at the level of collective political action.

Mill's second question concerned why we should bother to scrutinize opinions that are known to be true. Of course such "known" truths do sometimes turn up false or incomplete on close examination. As in the case of debate about false opinions, this improvement in our beliefs is again the best reason to encourage debate. In the many cases where debate simply reinforces our confidence that the opinion is true, though, Mill still insisted on the importance of open debate. Those impatient with endless talk about such "obvious" matters might reasonably ask why we should not just accept established truths at face value. Mill's response is that

> this is not the way in which truth ought to be held by a rational being. This is not knowing the truth. Truth, thus held, is but one superstition the more, accidentally clinging to the words which enunciate a truth. (1859)

The word "superstition" here indicates a kind of servitude, a situation in which beliefs and actions are determined not by one's own will, but rather by external factors beyond one's present understanding–and hence mysterious. Such servitude to unknown or incomprehensible influences is the very opposite of autonomy.

The principle of equality likewise enters into Mill's defense of unrestricted discussion. He recognized that in most controversial matters the question of truth and falsity is far from clear. Usually neither side possesses the whole truth *or* the whole error. In such cases, Mill observed, "When there are persons to be found who form an exception to the apparent unanimity of the world on any subject, even if the world is in the right, it is always probable that dissentients have something worth hearing to say for themselves, and that truth would lose something by their silence." The principle of considering any opinion whatever implies "giving merited honour to every one, whatever opinion he may hold, who has calmness to see and honesty to state what his opponents and their opinions really are…" (1859). In brief, Mill presents Locke's principle of equality so as to involve both equal access to the best available information, and equal opportunity to influence public opinion and action, in whatever area an individual feels inclined to participate.

These, then, are the most basic political ideals of classical liberalism:

- Autonomy
  - Self-determination of personal belief and action
  - Free choice among live alternatives for belief and action
  - Informed consent to others' actions affecting oneself
- Equality
  - Access to the means for attaining basic goods (*e.g.* life, liberty, property, pursuit of happiness)
  - Opportunity actually to attain such goods

My claim is that both the OS/FS movement and the culture that generated it are products of institutions shaped by these Enlightenment ideals. It is no accident, for example, that the Free Software Foundation originated at a place like the Massachusetts Institute of Technology rather than elsewhere. The few thousand key individuals behind the movement generally embrace these Enlightenment ideals, and the movement itself tends to advance both these ideals and the social institutions in question. As for Mill and Locke, they would be quite baffled by our technology but not by the hacker slogan "Information wants to be free!"

The institutions I have in mind, and some of their notable features, include:

- Democratic governance

  Bills or declarations of rights
  Universal suffrage
  Open meeting policies & sunshine laws
  Freedom of information laws
  Due process & trial by jury rather than secret police

- Education

  Mandatory public education
  Subsidized access to higher education
  Public libraries
  Literacy programs
  Liberal education rather than vocational training

- Science

  Publicly funded research and technology
  "Pure" research
  Blind peer review
  Falsification & fallibilism
  Shared results rather than trade secrets

- Free markets

  Open markets
  Merit scale
  Equal pay for equal work
  Entrepreneurial spirit
  Laissez faire regulation rather than central planning

Some of the listed institutional features (publicly funded science and technology) obviously contributed more to the emergence of the OS/FS movement than others (trial by jury). And we can certainly point out myriad ways in which the governments, schools, research centers, and businesses we know fall short of the classical ideals. My claim, though, is merely that *at their best* these institutions embody Enlightenment ideals, that the OS/FS movement developed out of such institutions, and that these ideals are manifest in that movement.[5] With this overview

[5]The free market is the most problematic of the four institutions mentioned in this connection. There is a sometimes stunning gap between classical liberal ideals and illiberal reality in modern business. One difficulty is that virtually *all* modern views of business trace back to conflicting inter-

of the historical influence of classical liberalism in mind, we are prepared to consider the social values at stake in closed *vs* open models of software development and distribution.

2 Contemporary Implications

Eric Raymond takes magic as his trope in one essay on OS/FS, quoting a wonderful line from Arthur C. Clarke: "Any sufficiently advanced technology is indistinguishable from magic" (Raymond 1999, 139). Herein lies a major political danger of advanced computing: reliance on magic in any form is antithetical to Enlightenment principles. Karl R. Popper began and ended volume 1 of *The Open Society and Its Enemies*, written in Europe at the outset of World War II, with references to magic that ought to give us pause. He began the book by locating the birth of European civilization at the point of "transition from the tribal or 'closed society', with its submission to magical forces, to the 'open society' which sets free the critical powers of man." He concluded it by saying, "Once we begin to rely upon our reason, and to use our powers of criticism, once we feel the call of personal responsibilities, and with it, the responsibility of helping to advance knowledge, we cannot return to a state of implicit submission to tribal magic" (1966, 1, 201).

To illustrate the connection I am getting at among magic, politics and software, imagine a people who do not understand how and where their food is grown, their clothes are made, their laws are formed, or their news is communicated. These things then appear before them "as if by magic." But of course these important goods *actually* appear due to the efforts of some external authorities, perhaps entirely unknown, who are trusted implicitly. The situation suggests the following corollary to Clarke's statement: "Any sufficiently powerful authority is indistinguishable from magic." Whether the authorities mean well or are *worthy* of trust is beside the point, as is the question whether the people feel content in their ignorance and vulnerability. We are concerned here about their autonomy, and whether they could restore it through their own efforts. Everything depends on whether they could, if they chose, acquire a thorough understanding of the processes that create and deliver their basic goods.

We need hardly be reminded that computing has become more and more central to the creation of basic goods in "advanced" societies. Most citizens, including most well-educated citizens, though, presently regard computing as essentially magical. Here too everything depends on whether they could, if they wanted, acquire complete understanding of the computing processes that are key to creating certain of their basic goods. Proprietary software guarantees that they *cannot*; only

pretations of Locke's view of property. A thorough account would involve retrieving an older notion of "the professions" that predates the modern concept of "business."

the copyright holders, the "computing authorities," have access to the sources of this magic. In Popper's terms this is a clear mark of regression, not advancement. OS/FS, on the other hand, guarantees that they *can*; copyleft gives everyone the equal right of unrestricted access to source. Though it is many other things besides, the OS/FS movement is thus in fact a revolution: a few members of the public have created the means to resist authorities whose actions undermine our autonomy and equality.

Of course not everyone wants to attain complete understanding and control of their computing, just as not everyone wants to attain genuine understanding of their beliefs through the rigors of public discourse. The Enlightenment ideals support those individuals, however many or few, who will not blindly submit important matters of personal belief, action, or livelihood to the mysterious forces of authority. We come here to the hard bottom line of classical liberalism: certain things are essential to protecting autonomy and equality. In any society, the opportunity for unrestricted public discussion of ideas is one such thing; in a computer-reliant society, the opportunity for unrestricted access to some body of functional source code is another. OS/FS guarantees that control of computing "magic" is available to those who do want or need it.

There is far more to the political and social significance of OS/FS than this hard bottom line of principle; likewise, there are other approaches to software development and distribution that could accomplish the same ends. Rather than develop these additional themes on an entirely abstract plane, however, let us look for them as we survey the potential significance of OS/FS within our familiar four institutions.

2.1 Government

Citizens *must* interact with government in some situations, for example in filing tax returns, responding to legal actions, applying for permits, and learning about new laws and regulations. Other interactions, such as communicating with public officials, obtaining information about public meetings, organizing political parties or actions, or proposing ballot initiatives, are not *required* of citizens but may be important rights. A government has the duty in such cases to provide for roughly equal access to these interactions. This duty explains why courthouses in the U.S. are usually located within a day's horseback ride of the furthest county border, why public notices are printed in the local newspaper, why the justice system employs so many translators, and why public buildings are being made handicap-accessible.

The most visible attempt to extend this principle to the realm of electronic communications is a proposed French law mandating that all public agencies and organizations use "open standards" software (Finley 2000). The law's backers identify five constitutional principles as its basis: free access to public information, re-

trievability of public data, national security, consumer security, and interoperability (OSSLaw nd). It means little to open an information channel if only a limited number of people can obtain the tools to use it, or if those tools become unavailable in a few years due to a manufacturer's decision.

The proposed law also addressed both "national" and "consumer" (or citizen) security concerns. Consider the following illustration of what this means: several years ago I considered filing my state tax return electronically. At that time, in Michigan, I would have had to first purchase a proprietary tax-preparation program. I did not like the idea of paying $49 to a retail store for what was in effect a license to pay my taxes. Nor did I like the idea that neither the Governor nor I could confirm what would happen to the data that I was expected to put into this magician's hat. I bought stamps instead (not because I am paranoid about man-in-the-middle attacks on my data, but because it just seemed too expensive). It did occur to me, though, that the Governor ought to have considered what a slick man-in-the-middle could do, given an opaque application that solicits confidential tax information and puts it online. However rare these kinds of security breaches are, they can and do appear from time to time. The transparency of open source software all but eliminates them.

Another advantage of promoting OS/FS in government is that it gives the public more opportunity to influence the evolution of our public technological infrastructure. OS/FS invites inspection, critique, modification, and discussion of alternative software solutions. This in turn gives the public a hand in shaping technology design and policy. Reliance on closed solutions puts all the design decisions in a few hands, and design decisions accumulate to become policy decisions by default. It is now largely taken for granted, for example, that the registered licensee's identity will be automatically, obscurely, and often incorrectly embedded as the "author" of many office documents. Whether average users might have wanted more control over such disclosure of their apparent identity is simply irrelevant, because the designers of popular proprietary programs long ago decided it would be this way. The *de facto* policy result is that it is now presumed to be very difficult, and hence unusual or even suspect, to conceal one's identity when creating an ordinary letter as a computer file.

Citizens who already regard computing as magical may fall easily into accepting a particularly insidious form of economic and technological determinism. On this view, whatever technology we have is determined by fate, or "progress," acting through large corporations. Standard Oil founder John D. Rockefeller, head of the largest corporate monopoly in history, maintained that human progress is inevitable and that it advances in exactly this way. The corporations that produced the key technologies in Rockefeller's day held awesome powers of monopoly. Not surprisingly, Rockefeller saw this a *very good* thing: such power, he said, is the "'working-out' of a law of nature and a law of God" (Hickman 1990, 141). OS/FS

pulls back the curtain that such Oz-like monopolist wizards would draw across the inner workings of our technological infrastructure.

2.2 Education

Schools are beginning to discover the practical advantages of OS/FS. Chief among these advantages is the potential savings in licensing costs–a major consideration when equipping hundreds of workstations on a public education budget. Most of the commentaries on OS/FS in education focus on these practical factors, which does get administrators to listen. To persons interested in these arguments, I recommend the articles by John Hartzog (1999) and Jeff Covey (2000).[6] As an educator, though, I am much more interested in the observation that "Free software both encourages learning and experimentation and in turn benefits from it" (Yee 1999).

My own curiosity about OS/FS began when, in a review of university classes intended to teach mathematical and logical reasoning, I encountered a major software maker's logo on a syllabus. It is certainly possible to teach the structures of logic using a computer language such as visual basic. Other features of the syllabus made me suspect, however, that the course was designed merely to train students in using a popular commercial product. There is a difference between university education and vocational training, which has to do with the intended breadth and depth of understanding the student acquires. Some mastery of logic and critical thinking is, or ought to be, one mark of the well-educated person. Language, logic, and habits of critical thinking–the great egalitarian tools of public discourse–constitute the original public information technology. They carry no trademark, copyright or patents and are available to any who want to learn them. A student who has learned logic has already learned the hard part of programming. Conversely, a student who learns the general *principles* of programming (not just how to do a few exercises in a specific programming environment) has also learned a great deal of logic and critical thinking. Students *could* learn general principles using open and fee-free programming languages, then transfer that understanding to proprietary languages when necessary. David Bollier offers the following explanation of why things are not done this way:

> many major universities allow themselves to serve as marketing and recruitment vehicles for Microsoft, accepting discount software deals in return for exclusive access to a future consumer base and programmers' mindshare. But even these seemingly smart deals can prove to be more expensive over the long term as licensing regimes are changed to extract higher prices once the company has a monopoly stranglehold. (1999, III. B. 2.)

[6] I am presently building a free software server for my own course web sites, to demonstrate the possibilities of this approach.

Even if Microsoft does this, we should recall that the strategy was pioneered by Apple. My question is whether we are trying to develop our students' intellectual skills, or merely their brand loyalty.

OS/FS is an even more attractive educational tool when we turn from general education to the computer and information technology curriculum specifically. Bollier again gets at the key point:

> its inner logic - the source code - can be directly manipulated by students. With its inner parts visible, users can choose to learn how the software works, and then share and develop that knowledge. Proprietary software, by contrast, is inherently "unknowable" because its inner architecture is a trade secret. (1999, I.E.)

Education ought to equip students to become self-teachers, and this has as much or more to do with developing attitudes and habits of thought as with acquiring skills and information. The American philosopher John Dewey warned about substituting vocational training for education. Mere training produces servile workers who "do what they do, not freely and intelligently, but for the sake of the wage earned. It is this fact which makes the action illiberal, and which will make any education designed simply to give skill in such undertakings illiberal and immoral" (1944, 260). We can teach both skills and understanding using open source software. This route also teaches that knowledge is not a branded commodity, an idea that ranks high on the list of *other* things that students need to learn in order to be free citizens.

2.3 Science

The editors of the essay collection *Open Sources* observe in their introduction that science "is ultimately an Open Source enterprise. The scientific method rests on a process of discovery, and a process of justification. For scientific results to be justified, they must be replicable. Replication is not possible unless the source is shared: the hypothesis, the test conditions, and the results" (Dibona et al. 1999, 2). It was in the spirit of sharing rather than hoarding the results of his work that Stallman developed his concept of Free Software and the GPL (1999a, 53-56). During the last decade there seems to have been, in teaching and research, a movement away from "computer science" and toward "information technology." This trend may reflect the proprietary model's influence on computer professionals' thinking. Science is at bottom a search for understanding, and is characterized by an ethic of open experimentation and shared results. Technology development is above all a business enterprise, typified by non-disclosure and intense competition for external

rewards.[7] The unlimited flexibility and control offered by OS/FS opens up avenues of experimentation that are otherwise unavailable. OS/FS has allowed many computer professionals to conduct at least some of their work in a collegial and scientific spirit once again.

Open source also invites critical peer review, another essential feature of science that is foreclosed by proprietary models. Paul Vixie describes its role in software development:

> An additional advantage enjoyed by open-source projects is the "peer review" of dozens or hundreds of other programmers looking for bugs by reading the source code rather than by just executing packaged executables. Some of the readers will be looking for security flaws and some of those found will not be reported (other than among other crackers), but this danger does not take away from the overall advantage of having uncounted strangers reading the source code. These strangers can really keep an Open Source developer on his or her toes in a way that no manager or mentor ever could. (1999, 98-99)

OS/FS is constantly used, reviewed, debugged and extended by users who quite often give their patches back to the community for free. An OS/FS slogan says "Given enough eyes, all bugs are shallow" (Raymond 1999, 41). This ongoing process of testing and revision has made many OS/FS applications extraordinarily powerful, stable and reliable. Such applications as the Apache web server and the Perl scripting language demonstrate how complex software engineering projects can be accomplished by using OS/FS to leverage the scientific ideals of extensive peer review and the spirit of common inquiry.

Cryptography is clearly the pre-computing scientific area where such peer review of source code has borne the most fruit. Modern cryptography is almost entirely computer-based. The calculations involved are so involved that only a machine can reliably carry them through. Conversely, the wide availability of computing power has permitted innovations in cryptography that would otherwise have been impossible. While mathematical symbols are still the central language of theoretical cryptography, source code *is* the language of applied cryptography. The only acceptable test of a modern cryptographic system is the pragmatic one: make the source code available, then wait for the world's experts to identify weaknesses in the algorithm and its software implementation. Many of the ongoing discussions on the Usenet group sci.crypt could be used as textbook illustrations of the scientific method in action. Other scientific disciplines that rely heavily on computing, such as some branches of mathematics and statistics, are also experiencing a sim-

[7] A parallel division has arisen between medical science on one hand, and the health care, pharmaceutical, and genetic engineering industries on the other.

ilar transition.[8] Whenever computer programs cease to be mere tools for doing research and become the research result itself, source code must be made available.

In most scientific disciplines, computing is still just a tool. It is hard to imagine a scientist, though, who would not prefer to have as much understanding and control as possible when it comes to important research tools. OS/FS is increasingly favored for research applications, for the simple reason that it permits unrestricted customization and troubleshooting. Scientists have always modified their physical tools.[9] Software is different, though. Consider just two examples. The first is the data system assembled and installed on a NOAA "hurricane hunter" aircraft in 1998. This complex amalgamation of radar equipment and laptop computers illustrates the challenge that can be involved in building "customized research equipment." Two of the team members, C. Wayne Wright and Edward J. Walsh, reported that "Thanks to the reliability of Linux and all of the off-the-shelf real-time data processing programs available in that domain, we were able to put together a state-of-the-art data system on a very tight schedule with a great variety of real-time displays" (1999, 30). The second is the development of Beowulf clusters, NASA's "off-the-shelf supercomputer" parallel processing project. In an initial progress report on this project we find the following rationale for using OS/FS wherever possible:

> The cost issue is important because to pay for an [operating system] license for each node of a multihundred-node beowulf cluster could be prohibitively expensive. Source code availability is important because it enables custom modifications to facilitate parallel computation. (Sterling et al. 1998, 2)

Without the source, no amount of desire or will would have permitted the kinds of fundamental software alterations involved in these two projects.

Modern science is an Enlightenment invention. It has always flourished best in an environment that places a premium on autonomy and equality. One last observation from a Beowulf developer will highlight the importance of these ideals for science. After describing programmers' years of frustration "fighting" with software vendors and system administrators, he says that the turn to OS/FS ushered in an entirely new attitude among researchers: "The realization is that learning to build and run a Beowulf cluster is an investment; learning the peculiarities of a specific vendor only enslaves you to that vendor" (Merkey 1998).

[8] The recently discovered solution to Fermat's Last Theorem represents one such case, where the source code can be considered an integral part of the research result.

[9] No chemistry lab is complete without a good glass-worker, and researchers regularly violate patents and warranty terms in the quest to improve their research equipment.

2.4 Business

In a nutshell, an aggressive business seeks to do two things. First, it wants to develop a competitive edge through product innovation, superior service, cost efficiency, and other factors. Second, it wants to keep that edge as long as possible so as to continue generating profits. It is not hard to understand how the reduced cost, increased control, functional transparency, stability, and flexibility of OS/FS may help a business establish and maintain a competitive advantage. Not surprisingly, a growing number of businesses, across the whole economy, are embracing OS/FS.

Three potential effects of OS/FS on business are particularly worth examining, however. The first concerns the issue of security. In any industry, *information* is essential to running a successful business. In certain industries, information is itself the chief commodity. Where a business manages its key information with computers, the managers must be concerned about security. Corporate espionage, computer crackers and vandals, even cyber-terrorism may present threats to a business. OS/FS can in principle offer greater security than proprietary software. Mature, widely used OS/FS applications are typically quite stable, for one thing. For another, access to source code makes it virtually impossible for hostile parties to insert trojan horse code or backdoor access that would compromise a system. Simon Cozens explained the principle very clearly in a comment posted to Technocrat.net:

> If I cannot see how the software on my machine works, I am surrendering my computer, in trust, to the vendor. Putting binaries onto my computer is not just opening the door to strangers–it's giving them the spare room. I don't do it for my house, and I wouldn't do it for my computer either. (2000)

Open source code offers an unmatched level of security, as long as businesses actually *use* the advantages provided by complete transparency.[10]

A second major business effect is specific to the software industry. The emergence of OS/FS has begun to redefine the notion of what a software company does. Traditionally, most businesses manufactured and sold *things*. Software companies naturally tended to think they should do something similar. Metaphysically speaking, though, software is an altogether different kind of entity from a product like shoes. The shoe store knows it is providing physical objects and some service in return for payment. The software company, on the other hand, has to consider carefully what it *is* selling. It might appear to be floppy disks and CD-ROMs, existing

[10] Elias Levy correctly points out that "simply being open source is no guarantee of security" (Levy 2000). The security advantage is only realized if qualified experts actually review the source code, and this trusted source is then used to build the executable programs. OS/FS is thus a necessary but not a sufficient condition for increased security.

programs, support services for existing programs, consulting and development services, or some combination of these. The physical objects in question, such as disks and CD-ROMs, are nearly worthless and easily duplicated. An existing program is nothing more than a very large, unique binary number that can be represented in a variety of media. As Donald Knuth, the mathematician and author of the free typesetting program TEX has pointed out, a number may cost a lot to discover–but once discovered, it isn't usually considered to be the sort of thing anyone can own (Advogato 2000). Accordingly, product support, development and consulting are emerging as the real business of the software industry. Companies like Red Hat Linux have embraced this business model wholeheartedly (Young 1999). By devoting itself entirely to OS/FS, such a company is never tempted to think that it is in the business of selling valueless plastic disks or priceless eternal numbers. By giving the software away, to as many people as possible, Red Hat and other OS/FS-oriented businesses create a large base of potential customers for their support services.

The last implication of OS/FS for business is one that has yet been scarcely recognized at all. Open source all but eliminates the concept of the "trade secret," "proprietary formula," or "patented law of nature" as the source of a competitive edge. What is left to create an edge is simple *excellence* in the goods and services provided. If I can choose among five different free programs to create my electronic documents, for example, I will likely choose the one that works best, has the most useful features, is the most reliable, allows the easiest information exchange with other users, and has the best support. If the people who produce and support these programs want my business, they will work very hard indeed to out-pace their competitors. This is the kind of competition that benefits the public and rewards those who are the best at what the industry does, which is to write software and support users. The proprietary model, on the other hand, can encourage behavior–such as development of standard-breaking and inefficient (but attractive looking) software, deliberately misleading advertising, and underhanded sabotage or destruction of competitors–that ultimately harms the public and undermines a company's own strengths. The proprietary model of development and distribution does usually reward those who excel at writing and supporting software, but it also tends to reward those who are the best at these other counterproductive and unfair business practices. Worst of all, the proprietary mode of competition can make such business practices appear very attractive, even to people of high integrity.

This aspect of the OS/FS movement may already be migrating to non-software industries. Researchers at the Rocky Mountain Institute's Hypercar Center have developed technology that they hope will revolutionize automobile design. This technology might be worth a very large fortune, but Research Director Amory Lovins has chosen to initiate a market experiment with it:

> Rather than following the traditional route of patenting and auctioning the intellectual property and hoping the buyer would succeed with it and not sit on it, the Center chose–by analogy with the open-software development model–to put most of its intellectual property in the public domain and get everyone competing to exploit it. This was expected to increase the number, quality, and motivation of market actors and to speed their progress. (Lovins nd)

If we hold to the principle of open access to information, we should advocate similar "open sourcing" of other advanced technologies such as genetic crop modification and medical technology. Extensive peer review, experimental control, and thorough understanding of the mechanisms at work in these process are all needed if we are to determine the wisest use of such technologies.[11]

2.5 Sacred Source, Blessed Binaries, Non-Free Distribution, and Limited-Access Source

The OS/FS movement is still in its early stages. OS/FS offers numerous benefits, but not all of these benefits are relevant or desired in all situations. Consider the three following variations on Free Software as current examples only. Further adaptations of the GPL will certainly proliferate. *1.* It has proved impractical to allow user modifications of the Distributed.net client software, since the project requires the clients to process calculations identically on thousands of computers and report the results back according to a standard protocol (Anonymous 2000a). At the same time, the owners of all these machines ought to know exactly what code they are volunteering to execute. Such applications can be distributed as "sacred source" code. The bulk of the source is available for inspection and testing, but executable binary programs built from it cannot interact with the central system. Only "blessed binaries" obtained from a central distributor are fully functional. The distributor in turn assumes responsibility for the integrity of the executables: they are implicitly guaranteed to be safe and legitimate programs. *2.* The major issue in the Free Software *vs* Open Source rift has to do with restrictions on distribution. The GNU Public License guarantees all users the right of unlimited distribution, while the Open Source Definition allows certain restrictions of these rights. Social activists,

[11] Open Source advocate Bill Joy has recently argued that certain emerging technologies pose an unprecedented danger to society at large, because the material and financial barriers to their (mis)use are minimal: "Knowledge alone will enable the use of them." In such a situation it appears that either the knowledge ought to be kept secret, so as not to fall into dangerous hands, or else the knowledge ought never to be sought in the first place. Joy endorses the latter course, arguing that research into robotics, genetic engineering, and nanotechnology ought to be limited or relinquished altogether (2000). The principles behind OS/FS entail only that *if* such knowledge is acquired, it ought to be made openly available.

educators and researchers tend to prefer the GPL, while many businesses are attracted to the OSD (Free Software Foundation 1991; Perens 1997). *3.* In the face of the French "Open Standards" proposal, proprietary software companies are reportedly considering a number of different strategies that would allow them to continue selling in that country (Anonymous 2000b). Such companies could begin using more standard, interoperable data formats in their programs. It may be possible to set up a "limited-access source" arrangement that would allow government officials to inspect and hold the source, but prohibit them from distributing it or disclosing trade secrets until the vendor either gives permission or goes out of business.

OS/FS is a hot new paradigm, but paradigms always beget variants. I have here mentioned only three of many possible variations on the OS/FS concept. We know from other industries, like transportation, that the market will support a wide array of similar products. Some people pride themselves on never paying more than $500 for a car, while others have the means and desire to drive nothing but a new Jaguar. Much current Open Source software is the computing equivalent of the Jaguar,[12] but we are already seeing both "free" and "professional" or "premium" releases of some open source software.

Let people decide for themselves what to drive and what software to run, but let us insist on the kind of variety and genuine *choice* that OS/FS currently provides. Just as the hard bottom line of classical liberalism is satisfied as long as there is effective public transportation in a mobile society, so is it satisfied if there is effective public software in an information society. At the moment there is, and it happens to be excellent software. Those who consider the classical liberal ideals important will provide the OS/FS movement the support and protection it needs to stay that way.

Postscript: Some Weak Objections to the OS/FS Model

My intent in this paper is to examine the historical and philosophical heritage of OS/FS, and to present an advocate's view. I hope others will engage this debate with responses and offer the kind of counter-arguments to OS/FS that will help us thoroughly understand the movement's implications. To prepare the way for such debate, consider the following sketch of several common but manifestly *weak* objections to OS/FS, together with an advocate's brief responses.

[12] Neal Stephenson uses a more extended transportation analogy to make this same point. If the Microsoft Windows operating systems are like family station wagons, their GNU/Linux counterparts are like tanks "made of space-age materials and jammed with sophisticated technology from one end to the other. But they are better than army tanks. They've been modified in such a way that they never, ever break down, are light and maneuverable enough to use on ordinary streets, and use no more fuel than a subcompact car" (1999).

OS/FS undermines capitalism?

The concern is sometimes expressed that the OS/FS model is detrimental to the proprietary model and the businesses that rely on it.

Objection. It is ethically legitimate to profit from proprietary software.

Response. Indeed it is, and OS/FS does not exclude others who find a market for products developed and distributed under the proprietary model. OS/FS is an alternative to proprietary software, not an attack on its existence. Proprietary software distributors who perceive OS/FS as a threat should recall that there is no guarantee that any particular kind of business will always remain profitable, though: ice houses went bankrupt once people were able to buy their own refrigerators.

Objection. OS/FS threatens the legal right to copyright, patent or otherwise protect one's intellectual property.

Response. An OS/FS license can only be granted by an author, and only to his or her own works. My intellectual property rights are not infringed by your decision to release your creation under such a license.

Objection. OS/FS threatens the place of the profit motive as an incentive to innovate and to improve software products.

Response. This may be true. It is hard to imagine that anyone will get rich in the near future by writing a better web server than what is already available under the GPL. On the other hand, the profit motive is usually overrated by those who are unfamiliar with the non-monetary rewards of work and competition. The OS/FS approach simply appeals to other persons, and other motives. Eric Raymond argues throughout *The Cathedral and the Bazaar* that the OS/FS model does motivate innovation and advancement (1999).

OS/FS undermines software uniformity?

Centrally controlled of software development (the "cathedral" model) ensures that commonly used programs employ uniform standards, file formats, and user interfaces; decentralized development (the "bazaar" model) promotes proliferation and wide differences among programs, resulting in the computing equivalent of the Tower of Babel.

Objection. Compatibility & Interoperability, or, "That guy in the next office says he can't read my email attachments!" Proprietary software like Microsoft

Office, Adobe Photoshop and Lotus cc:Mail are the standard means for information exchange in many organizations. OS/FS users are often unable to access files their colleagues create with these proprietary applications.

Response. Compatibility and interoperability depend on the use of common software standards. OS/FS relies on open standards, while much commercial software relies entirely or partly on proprietary standards. Open formats such as ASCII text for email and system files, hypertext markup language (html) for web documents, portable document format (pdf) for paper publications, and numerous open image and audio file formats (including jpg, tiff, wav, and mp3) promote compatibility across different applications and hardware platforms. Common software libraries like the GNU ToolKit and the GNU C libraries promote software interoperability. Open standards are usually adopted by OS/FS developers after general discussion of their technical merits. When a consensus emerges on the "best" standard, it becomes the agreed-upon foundation for future software development.

Proprietary standards are sometimes created from scratch to serve particular purposes, but are often simply adaptations or modifications of existing open standard formats. In either case, their internal specifications are usually kept as trade secrets so that few applications from other developers, if any, can access them effectively. Users thus often find themselves wedded to a software brand not because they need the additional features, but because they do not know how to convert existing files into more accessible formats.

The scenario described in the objection is all too common: users assume that everyone in the world relies on identical software, and hence on identical standards. This is not greatly different from assuming that everyone in the world speaks one's own language, except that nobody has to buy a license to learn English or Spanish. Like particular languages, proprietary standards can offer unique advantages, but they also involve several disadvantages. Where the proprietary standard offers needed functionality that is otherwise unavailable, an organization will find that it must provide all involved parties with the appropriately identical software and hardware. Moreover, vendors have a financial incentive to introduce frequent upgrades that add functionality but render previous versions of their software obsolete. Such changes actually erode compatibility: If one key person insists on using software based on the new standards, everyone else must either pay the "upgrade tax" of a new program license or else drop out of contact. This regular upgrade cost appears especially high if the vast majority of users never take advantage of the "improved" functionality. The result is that many desktop computers are now bloated with the latest available industrial-grade multimedia technology, which is in many cases used mainly to read plain ASCII email.

On the other hand, very few users seem to realize that most programs offer a "Save as…" option, which can put files in more accessible open formats. The guy in the next office actually *could* read those beautiful attachments we create in the latest version of Office or WordPerfect–all that is necessary is to take a few extra seconds to put them in an appropriate format. Perhaps we should ask whether we are trying to communicate with one another, or to win a contest to see who has the most "advanced" magic?

References

Advogato. "Interview: Donald E. Knuth." Advogato.org. Electronic document. http://www.advogato.org/article/28.html. 25 January, 2000. Accessed 26 January 2000. 15

Ananian, C. Scott. "Criminal Code?." Salon. Electronic document. http://www.salon.com/tech/feature/2000/02/09/linuxdvd/print.html. 2000. Accessed 10 February 2000. 2

Anonymous. "FAQ: Frequently Asked Questions on the Microsoft Antitrust Case." Center for the Moral Defense of Capitalism. Electronic document. http://www.moraldefense.com/Campaigns/Microsoft/Antitrust_FAQ/default.htm. 1999. Accessed 1 December 1999.

—. "Distributed.net Source." Distributed.net. Electronic document. http://www.distributed.net/source/. 24 May, 2000. Accessed 30 May 2000. 16

—. "MS: 'Not So Fast There, Pierre'." Wired. Electronic document. http://www.wired.com/news/print/0,1294,35898,00.html. 25 April, 2000. Accessed 26 April 2000. 17

—. "Open Source: Software Gets Honest." OpenSource.org. Electronic document. http://www.opensource.org/. nd. Accessed 15 December 1999.

Bollier, D. "The Power of Openness: Why Citizens, Education, Government, and Business Should Care about the Coming Revolution in Open Source Code Software." Berkman Center for Internet and Society. Electronic document. http://eon.law.harvard.edu/opencode/h2o/. March, 1999. Accessed 17 May 2000. 10, 11

Covey, Jeff. "Linux in Education." Freshmeat. Electronic document. http://freshmeat.net/news/2000/03/18/953441940.html. 18 March, 2000. Accessed 19 March 2000. 10

Cozens, Simon. "Re: Open Source Critique Criticized." Technocrat.net. Electronic document. http://technocrat.net/955986079/index_html. 17 April, 2000. Accessed 18 April 2000. 14

Daffara, C. et al. "Free Software / Open Source: Information Society Opportunities for Europe?." Working Group on Libre Software, European Commission. Electronic document. http://eu.conecta.it/. 1999. Accessed 15 December 1999.

Dewey, John. *Democracy and Education*. New York: The Free Press, 1944. 11

Dibona, Chris, Sam Ockham, and Mark Stone, editors. *Open Sources: Voices from the Open Source Revolution*. Sebastopol, California: O'Reilly & Associates, Inc., 1999. 11, 22, 23

Finley, Michelle. "French Pols Say, 'Open It Up'." Wired. Electronic document. http://www.wired.com/news/print/0,1294,35862,00.html. 24 April, 2000. Accessed 26 April 2000. 8

Free Software Foundation. "GNU General Public License, Version 2." GNU Project. Electronic document. http://www.gnu.org/copyleft/gpl.html. 1991. Accessed 12 May 2000. 1, 17

Ghosh, R. A. and V. V. Prakash. "The Orbiten Free Software Survey." Orbiten Research. Electronic document. http://orbiten.org/ofss/01.html. May, 2000. Accessed 11 May 2000. 2

Hartzog, John. "Public Software for Public Education." The Web Project at California State University, Northridge. Electronic document. http://www.vcsun.org/~john/freeware.html. April, 1999. Accessed 24 April 2000. 10

Hickman, Larry A. *John Dewey's Pragmatic Technology*. Bloomington, Indiana: Indiana University Press, 1990. 9

Joy, Bill. "Why the Future Doesn't Need Us." Unpaginated offprint. *Wired* (April 2000). 16

Kaminsky, Dan. "Core Competencies: Why Open Source Is The Optimum Economic Paradigm for Software." Doxpara Research. Electronic document. http://www.doxpara.com/core.html. 2 March, 1999. Accessed 29 May 2000. 3

Lessig, Lawrence. "Code and the Commons.". Electronic document. http://cyber.law.harvard.edu/works/lessig/fordham.pdf. 9 February, 1999. Accessed 7 February 2000.

Lessig, Lawrence. "Open Code and Open Societies: Values of Internet Governance." Available as electronic document. <http://cyber.law.harvard.edu/works/lessig/final.pdf>. Accessed 15 December 1999. *Chicago-Kent Law Review* 74 (1999): 101–116.

Lettice, John. "French Proposal Plans State Free Software Agency." The Register. Electronic document. http://www.theregister.co.uk/000104-000005.html. 4 January, 2000. Accessed 12 January 2000.

Levy, Elias. "Wide Open Source." SecurityFocus.com. Electronic document. http://www.securityfocus.com/templates/article.html?id=19&_ref=96205544. 16 April, 2000. Accessed 17 April 2000. 14

Locke, John. *Second Treatise of Government*. Available as a Wiretap electronic document. <http://sunsite.berkeley.edu/~emorgan/texts/philosophy/1600-1699/locke-second-117.txt>. Public domain edition. 1690. 4

Lovins, Amory. "A Brief History of the Hypercar Center." Hypercar Center. Electronic document. http://www.hypercarcenter.org/dox/center_a.html. nd. Accessed 20 January 1000. 16

McHugh, Josh. “Open Sourcery.” Forbes.com. Electronic document. http://www.forbes.com/Forbes/99/0503/6309054a.htm. 3 May, 1999. Accessed 8 February 2000.

Merkey, Phil. “Beowulf: Introduction and Overview.” Center of Excellence in Space Data and Information Sciences. Electronic document. http://beowulf.gsfc.nasa.gov/intro.html. 11 September, 1998. Accessed 8 February 2000. 13

Mill, John Stuart. *On Liberty*. Available as a Wiretap electronic document. <http://sunsite.berkeley.edu/~emorgan/texts/philosophy/1800-1899/mill-on-215.txt>. Public domain edition. 1859. 4, 5

OSSLaw. “Open Standards & Source Code Access.” OSSLaw.org. Electronic document. http://www.osslaw.org/pr_en.html. nd. Accessed 26 April 2000. 9

Perens, Bruce. “The Open Source Definition.” Open Source Initiative. Electronic document. http://www.opensource.org/osd.html. 1997. Accessed 12 May 2000. 2, 17

—. “Open Source Critique Criticized.” Technocrat.net. Electronic document. http://technocrat.net/955986079/index_html. 17 April, 2000. Accessed 18 April 2000.

Popper, Karl R. *The Open Society and Its Enemies*. 5th edition. Volume 1 . Princeton, N.J.: Princeton University Press, 1966. 7

Raymond, Eric S. *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. Sebastopol, California: O’Reilly & Associates, Inc., 1999. 2, 7, 12, 18

Raymond, Eric S., Lawrence Lessig, et al. “Controversy: Should Public Policy Support Open-Source Software?.” American Prospect. Electronic document. http://www.prospect.org/controversy/open_source/. 1999-2000. Accessed 17 April 2000. 3

Rubini, Alessandro. “Software Libre and Commercial Viability.” *Linux Journal* (February 1999): 46–48.

Stallman, Richard M. “The GNU Operating System and the Free Software Movement.” See Dibona, Ockham, and Stone (1999), 1999. 53–70. 11

Stallman, Richard M. “Why “Free Software” Is Better Than “Open Source”.” GNU Project. Electronic document. http://www.gnu.org/philosophy/free-software-for-freedom.html. 6 November, 1999. Accessed 12 January 2000. 3

—. “Why Software Should Not Have Owners.” GNU Project. Electronic document. http://www.gnu.org/philosophy/why-free.html. 6 November, 1999. Accessed 12 January 2000. 3

Stephenson, Neal. *In the Beginning Was the Command Line*. New York: Avon Books, Inc., 1999. 17

Sterling, Thomas, Don Becker, et al. “An Assessment of Beowulf-class Computing for NASA Requirements.” Loki - Commodity Parallel Processing. Electronic document. http://loki-www.lanl.gov/papers/ieee_aero98/p312.ps. 1998. Accessed 8 February 2000. 13

Vixie, Paul. “Software Engineering.” See Dibona, Ockham, and Stone (1999), 1999. 91–100. 12

Wright, C. Wayne and Edward J. Walsh. “Hunting Hurricanes.” *Linux Journal* (February 1999): 20–30. 13

Yee, Danny. “Development, Ethical Trading, and Free Software.”. Electronic document. http://danny.oz.au/freedom/ip/aidfs.html. 30 November, 1999. Accessed 12 May 2000. 10

Young, Robert. “Giving It Away: How Red Hat Software Stumbled across a New Economic Model and Helped Improve an Industry.” See Dibona, Ockham, and Stone (1999), 1999. 113–125. 15

Typeset in LaTeX 2ε http://www.tug.org
on GNU-Linux http://www.linux.org