Chicago-Kent Law Review

Volume 74
Issue 3 *Symposium on Revised UCC Article 9*

Article 17

June 1999

Open Code and Open Societies: Values of Internet Governance

Lawrence Lessig

OPEN CODE AND OPEN SOCIETIES: VALUES OF INTERNET GOVERNANCE

LAWRENCE LESSIG*

Pierre de Fermat was a lawyer and an amateur mathematician. He published one paper in his life—an anonymous article written as an appendix to a colleague's book. But while he published little, he thought lots about the open questions of mathematics of his time. And in 1630, in the margin of his father's copy of Diophantus' *Arithmetica*, he scribbled next to an obscure theorem ("$X^n+Y^n=Z^n$ has no non-zero integer solutions for N>2"): "I have discovered a truly remarkable proof which this margin is too small to contain."[1]

It's not clear that Fermat had a proof at all. Indeed, in all his mathematical papers, there was but one formal proof. But whether a genius mathematician or not, Fermat was clearly a genius self-promoter, for it is this puzzle that has made Fermat famous. For close to four hundred years, the very best mathematicians in the world have tried to pen the proof that Fermat forgot.

In the early 1990s, after puzzling on and off about the problem since he was a child, Andrew Wiles believed that he had solved Fermat's Last Theorem. He published his results—on the Internet, as well as other places—but very soon afterwards, a glitch was discovered. The proof was flawed. So he withdrew his claim to having solved Fermat's Theorem.

But he could not withdraw the proof. It was out there, in the ether of the Internet, and could not be erased. It was in the hands of many people, some of whom continued to work on the proof, even though flawed. And after extensive and engaged exchange on the Net, the glitch was undone. The problem in Wiles' proof was fixed. Fermat's Last Theorem was solved.[2]

* Jack N. and Lillian R. Berkman Professor for Entrepreneurial Legal Studies, Harvard Law School. Thanks to Hal Abelson, Scott Bradner, David Johnson, Henry Perritt, and Richard Stallman for comments on an earlier draft.

1. *See* <http://www-groups.dcs.st-and.ac.uk/~history/HistTopics/Fermat's_last_theorem.html>; *see also* AMIR D. ACZEL, FERMAT'S LAST THEOREM: UNLOCKING THE SECRET OF AN ANCIENT MATHEMATICAL PROBLEM 6 (1996).

2. *See* SIMON SINGH, FERMAT'S ENIGMA: THE EPIC QUEST TO SOLVE THE WORLD'S GREATEST MATHEMATICAL PROBLEM (1997).

* * *

There's something of a movement going on out there—called by some the Free Software Movement (as founder, Richard Stallman puts it, free in the sense of "free speech," not in the sense of "free beer"[3]), and by others, the Open Source Software Movement.[4] The differences are important, but so are the commonalties strong. Both aim for a world in which the fundamental software—the code—governing the Internet is software that is "open"—code whose source is available to all, to be taken, to be modified, and to be improved.

The arguments for open code are many; the reasons favoring it, different. Most argue its virtue is efficiency: that the product of this open development, like the product of any open society—this code that has revealed its flaws by revealing its source, and that has been improved by revealing its source—is more robust, more efficient, more reliable, code than the product of any closed process. Better code is the promise, and in a world where computers are as reliable as electricity in Italy, this indeed is a valuable promise.

But it is not my aim here to discuss its efficiency; my aim is its values.[5] My question in the few minutes that I can have your attention is this: Can we learn something from the values of the Open Source or Free Software Movement that would teach us something about Internet governance, and governance generally?

* * *

Governance: to many, this idea of Internet governance will seem quite odd. This weird interactive television that somehow got connected to wildly confused libraries—what could it mean to speak about governance here?

But I mean governance in a very general sense. If you want to set up a server on the World Wide Web, you must register and receive a

3. Richard Stallman is the founder of this extraordinary movement and is truly its constituting force. *See* Amy Harmon, *The Rebel Code*, NY TIMES MAG., Feb. 21, 1999, at 34; Andrew Leonard, *Maverick Richard Stallman Keeps the Faith—and Gives Bill Gates the Finger* (Aug. 31, 1998) <http://www.salonmagazine.com/21st/feature/1998/08/cov_31feature. html>.

4. The present keepers of the keys for this branch of the Free Software Movement are Eric Raymond and Bruce Perens, who founded opensource.org. *See* <http://www.opensource. org/board.html>.

5. I therefore completely agree with Stallman that the issues raised by this movement are primarily issues of value first. *See* Richard Stallman, *Reevaluating Copyright: The Public Must Prevail*, 75 OR. L. REV. 291 (1996); *see also* Lawrence Lessig, *The Limits in Open Code: Regulatory Standards and the Future of the Net*, 14 BERKELEY J.L. & TECH. 759 (1999).

name—a domain name—from an Internet registry, right now a company called Network Solutions.[6] This procedure is the product of governance in the sense that I mean. When you connect to a site on the World Wide Web, your machine transmits to the site on the Web an address—your Internet Protocol ("IP") address—so that the machine on the Web can find you in return.[7] This protocol is the product of governance in the sense I mean. When you connect to a site with this IP address, the IP address need not provide information to identify who you are; it can be dynamic rather than static; it can be a proxy rather than real—nothing requires that the other side learn anything real about you.[8] This is the product of governance in the sense I mean.

In each case, the governance at stake is in part a governance that has been brought about by a certain architecture in the Internet. It is the code's design that IP addresses are used to identify locations on the Net; other designs could have been chosen. It is the code's design that only the IP address is needed to connect to a site; a different design, requiring greater security, could have been selected. Thus in part, the governance that I mean is a governance brought about through code.[9]

But obviously, governance is not just code. It was not software that chose Network Solutions as the domain name registry—it was the United States government, by a contract that shifted the

6. *See* <http://www.netsol.com/nsi>.

7. On page 103, the MICROSOFT PRESS COMPUTER DICTIONARY (3d ed. 1997) defines a communications protocol as "[a] set of rules or standards designed to enable computers to connect with one another and to exchange information with as little error as possible."

8. *See* MICROSOFT PRESS COMPUTER DICTIONARY, *supra* note 7, at 387 (defining a proxy server as a computer that intercepts Internet traffic and has the ability to keep users from accessing outside Web pages); *see also id.* at 197 (defining firewall).

9. I am using the term "code" here far more loosely than software engineers would. I mean by code the instructions or control built into the software and hardware that constitutes the Net. I include within that category both the code of the Internet protocols (embraced within TCP/IP) and also the code constituting the application space that interacts with TCP/IP. Code of the latter sort is often referred to, in Jerome Saltzer's terminology, as code at the "end." "For the case of the data communication system, this range includes encryption, duplicate message detection, message sequencing, guaranteed message delivery, detecting host crashes, and delivery receipts. In a broader context the argument seems to apply to many other functions of a computer operating system, including its file system." Jerome H. Saltzer et al., *End-to-End Arguments in System Design*, *in* INNOVATIONS IN INTERNETWORKING 195 (Craig Partridge ed., 1988). More generally, this layer would include any applications that might interact with the Network (browsers, e-mail programs, file transfer clients) as well as operating system platforms upon which these applications might run.

In the analysis that follows, the most important "layer" for my purposes will be the layer above the IP layer. This is because the most sophisticated regulations will occur at this level, given the Net's adoption of Saltzer's end-to-end design.

responsibility from the late John Postel.[10] It was not software that established the protocols for the World Wide Web. It was a group of Internet decision-makers who put a recommendation for this design into circulation, and then recognized it as adopted.[11] These decision-makers were people; some of them are responsible to "the People"; their name was not Hal 2000.

Thus governance in the sense that I mean is a mix of the regulations of code and the regulations of bodies that regulate this code. It is both machine and man.

But these "regulators" regulate in ways that are very different. They are different from each other, and they are different from the regulations of real-space governments. We should understand this difference.

* * *

First, think a bit more about code—about the way that code regulates. Lawyers don't like to think much about how code regulates. Lawyers like to think about how law regulates. Code, lawyers like to think, is just the background condition against which laws regulate.

But this misses an important point. The code of cyberspace—whether the Internet, or a net within the Internet—defines that space. It constitutes that space. And as with any constitution, it builds within itself a set of values and possibilities that governs life there. The Internet as it was in 1995 was a space that made it very hard to verify who someone was; that meant it was a space that protected privacy and anonymity. The Internet as it is becoming is a space that will make it very easy to verify who someone is; commerce likes it that way; that means it will become a space that doesn't necessarily protect privacy and anonymity. Privacy and anonymity are values, and they are being respected, or not, because of the design of code. And the design of code is something that people are doing. Engineers make the choices about how the world will be. Engineers in this sense are governors.

10. *See* Rebecca Quick, *On-Line: Internet Addresses Spark Storm in Cyberspace*, WALL ST. J., Apr. 29, 1997, at B1.

11. *See* Walt Howe, *Delphi FAQs: A Brief History of the Internet* (last modified Oct. 24, 1998) <http://www.delphi.com/navnet/faq/history.html>.

* * *

Now I've been selling this line about how the code is a kind of law, and how the authors of the code—code writers—are a kind of governor. I've been selling this line for what seems to be a very long time—months, which in Internet time is a generation.[12] And the obvious implication of this line of argument is that we need to think more about how these governors govern: If the architecture of the Internet affects values as well as bits—if the architecture is a kind of law—then as in any law-making context, we should be asking, who are these law makers, and how do they make law? That if the code reflects values, then we should identify the values that come from our tradition—privacy, free speech, anonymity, access—and insist that this code embrace them if it is to embrace values at all. That we should look to the structure of our constitutional tradition, and extract from it the values that are constituted by it, and carry these values into the world of the Internet's governance—whether the governance is through code, or the governance is through people.

* * *

Joe Reagle is a researcher at the World Wide Web consortium—a consortium of Internet companies that work on standards and protocols for the Net.[13] He's a young researcher; he was a fellow at the Berkman center from where I now hail; but he's been around the Net for most of its life. He's a bit hard to recall sometimes—his hair tends to change colors fairly regularly—but he's the best techno-policy geek I know in this field, and he's one who has heard this argument of mine many times before.

And recently he said this to me:

> You lecture about the values implicit in our constitutional tradition; you argue we should carry these to cyberspace. What about the values implicit in the Internet's tradition? What about the values that are implicit in how it is governed? Why shouldn't we identify those, and carry those to real space?

I realized he was right. Ever the imperialist, ever the lawyer, I had proceeded on the assumption that real space would have something to teach cyber. But why not the other way round? Why wouldn't the governance of the Net have something to teach real

12. The sale comes to an end in my book, CODE AND OTHER LAWS OF CYBERSPACE (1999).

13. *See* <http://www.w3.org/People/Reagle>.

space?

I mean this lecture to be the first part of an answer to Reagle's challenge. What are the values in the Internet's governance just now, and how would they translate? And does this tradition tell us something about the relationship between this growing movement of open source software, and the future of Net governance?

* * *

First, let's talk a bit more about open source. I started with the story of Wiles' proof of Fermat's Last Theorem. If you need a model to understand—if a description is obscure unless you can relate it to something else, if you have been so completely bent by something called legal education that you can't conceive of an idea on its own—then let this model, or this picture, of Wiles' proof stand as a template. For this is just how open source software works: with an inspiration, handed over to the public, in an imperfect but promising form, which a public then can take up on its own and continue to work out. But with a promise, that what they produce with this product leaves this open part open.

The history of an operating system called GNU/Linux is a perfect, and perfectly timely, example. In this story, Fermat is played by a young Finn named Linus Torvalds.[14]

Linus Torvalds is not a lawyer; I don't know whether he is an amateur mathematician. But in 1991, he was an undergraduate at the University of Helsinki, and he wrote the kernel of an operating system ("OS"), and posted it on the Internet. It was a file that was 10k lines large. And it did—as "kernel" suggests—just a fraction of what we now associate with what marketers call operating systems.

People took up Linus' invitation. They took up his kernel, and began to work out its problems. They took the proof that Linus had sketched, and over time, by linking it with Richard Stallman's GNU operating system, they turned it into an operating system.[15] And not just any operating system. Rather, they took this kernel and turned it into an OS that some have called the single greatest threat to the dominance of the dominant desktop OS developer, Microsoft. That's

14. *See* RANDOLPH BENTSON, INSIDE LINUX: A LOOK AT OPERATING SYSTEM DEVELOPMENT (1996); Linus Torvalds, *The Linux Edge*, *in* OPEN SOURCES—VOICES FROM THE OPEN SOURCE REVOLUTION 101, 109 (Chris DiBona et al. eds., 1999).

15. *See* Richard Stallman, *The GNU Operating System and the Free Software Movement*, *in* OPEN SOURCES—VOICES FROM THE OPEN SOURCE REVOLUTION, *supra* note 14, at 53, 65-66.

not my judgment. That's the judgment of an internal Microsoft document, part of the "Halloween documents," that picked GNU/Linux, and the Open Source Software Movement, as "a significant near-term revenue threat to Windows NT."[16]

Now there is something incredible in this story—some parts not believable, some parts extraordinary. The idea that through this collective, essentially volunteer, effort, one of the most powerful operating systems on the planet could be developed is, to put it mildly, surprising. Through a process that Mike Godwin describes as Internet barn-raising, a product gets built.[17] And not just with this OS: for as Godwin recounts, there are many contexts where something similar to this gets done—where a common problem is placed in a common space, and people from around the world turn themselves to working, in parallel, on the problem.

The groundwork for this project had been laid many years before. Torvalds wrote a kernel, but that kernel was plugged into a framework that had been developed since 1985.[18] This was the work founded by Richard Stallman, and his Free Software Movement. Angered by AT&T's decision to make Unix proprietary, Stallman had begun a similar project which had similarly drawn thousands of Net barnraisers. Its aim was an operating system, GNU, but GNU foundered for want of a kernel. It was only because Torvalds had GNU—and the tools that GNU had produced—that he could plug Linux into GNU. One free software project building upon another, to build GNU/Linux.[19]

This idea—or ideal—of open source software is not limited to this OS, GNU/Linux. It extends to many of the core technologies that make the Net run. And this idea, or ideal, of putting into the commons one's work product—of giving away what one makes, with no guarantee of compensation—might all sound wildly 60s-ish, wildly idealistic: Marx applied to code. It sounds alien to our tradition, foreign to what has made our nation flourish.

Until one thinks again about science, and about the way science works. For basic science functions much the same—progress made

16. *See* <http://www.opensource.org/halloween>.

17. *See* MIKE GODWIN, CYBER RIGHTS: DEFENDING FREE SPEECH IN THE DIGITAL AGE (1998).

18. Or since the 1970s, if you count emacs.

19. And to be fair, there was an important role played by law as well. Stallman's real genius (from the perspective of a lawyer, and I apologize for that) was the GPL—a license that would use the power of copyright to guarantee that what was produced under the GPL not be removed from the commons.

and then given to the next generation.[20] Or until one learns something about how the Net was built. For post-communist prejudice notwithstanding, the fact is that most of the Internet is open source in just this sense, and certainly most of the Internet that most of us have anything to do with is open source in just this sense. And most of the growth in the reach of the Internet has come from its being open source in just this sense.

Think about this fact for a second. There have been communications networks since the late part of the last century. There have been computer networks for the past thirty years. But these telecommunications networks were primarily proprietary. These networks were built on the ideal that code, and protocols, be kept private. And so private they were. Networks clunked along at a tiny growth rate. Costs were high, participation, low. And single dominant actors got to control how the network would be used.[21]

But the Internet was built on a different model. It is no accident that it came from the research community of major universities. And no accident that it was pushed onto these communities by the demands of government. And no accident that it could not take off until the government broke up the AT&T monopoly that controlled the telephone lines in 1984. Once forced, researchers began to build for the Internet a set of protocols that would govern it. These protocols were public—they were placed in a commons, and none claimed ownership over their source. Anyone was free to participate in the bodies that promoted these common codes. And many people did. Anyone was free to take the protocols and build applications that use them. This was a barn-raising, in Godwin's terms, that built this Net.

Most of the core software that runs the Internet as it is just now is barn-raising code in just this sense—from free Apache servers that

20. *See, e.g.*, Rebecca S. Eisenberg, *Intellectual Property at the Public-Private Divide: The Case of Large-Scale cDNA Sequencing*, 3 U. CHI. L. SCH. ROUNDTABLE, 557, 561 (1996); Rebecca S. Eisenberg, *Public Research and Private Development: Patents and Technology Transfer in Government-Sponsored Research*, 82 VA. L. REV. 1663, 1667 (1996); Michael A. Heller & Rebecca S. Eisenberg, *Can Patents Deter Innovation? The Anticommons in Biomedical Research*, 280 SCIENCE 698, 700 (1998); Michael A. Heller, *The Tragedy of the Anticommons: Property in the Transition from Marx to Markets*, 111 HARV. L. REV. 621 (1998).

21. In a well known story, when a design for something close to the Net was proposed by Rand Researcher Paul Baran—in 1964—an executive at AT&T, then owner of the single telecommunications network in the United States, is said to have responded, 'First . . . it can't possibly work, and if it did, damned if we are going to allow the creation of a competitor to ourselves.' JOHN NAUGHTON, A BRIEF HISTORY OF THE FUTURE: THE ORIGINS OF THE INTERNET 107 (1999).

are still the dominant server on the Internet, to the SENDMAIL protocol that channels most e-mail on the Internet.[22]

But it may be the World Wide Web that is the best example of this. The protocols—the language—with which the Web gets built are some things called HTML and HTTP. These are public standards, public programming languages, which are developed and dedicated to the public's use. So too are the languages BASIC and C++ public in this sense. But the Web took this public-ness one step further. For not only is the code, or the language, public and free for anyone to learn; but so too is the specific use of the code public and free for anyone to use. If you take a program written in C++, and try to display it, it will look like gibberish. But if you are on a Web page and tell the browser to "reveal source," it will look like English—the source is always visible. HTML source is public; it is in the commons; anyone can take it, and use it, as they wish. And so far at least, browsers have been written to insist that this source is open. To insist that people can take this stuff, and copy it, and modify it, and use it.

The consequence of making the code like this has been the fastest growing network in our history. Nonproprietary, public domain, dedicated to the commons, indeed some might think, attacking the very idea of property—yet generating the greatest growth our economy has seen. As an engineer at an Internet software company recently said to me—"the 'ah-ha' for Open Source Software came to me when I realized, 'wait, open source is the Internet.'"

* * *

Which brings us back to Joe Reagle. Reagle has a description of how the Net works. He has a model, or a story, or a set of descriptions from which one might mine the values that are implicit in this design. And so let's do this that I said he should do with our constitution—let's identify these values, and see what value they bring.

Reagle's first value is a kind of anarchy: that the Net gets run not through some organized system of power, but through bottom-up control. As Net founder Dave Clark put it: "We reject kings, presidents and voting. We believe in rough consensus and running code."[23]

To us—lawyers, quasi-well-off sorts—this doesn't sound like a

22. *See* Chris DiBona et al., *Introduction*, *in* OPEN SOURCES—VOICES FROM THE OPEN SOURCE REVOLUTION, *supra* note 14, at 1, 5-6.

23. Dave Clark, IETF Credo (1992) <http://www.vtac.org/Tutorials/ietf_hx.html>.

promising start. It sounds to us like chaos. But the techno-anarchist is quick to lecture that our understanding of what "anarchy" is is, well, wrong. Anarchy is not about rulelessness. Anarchy is about bottom-up control.

Whatever. I'm not so much interested in the Net's theorizing about its nature; I don't really care for what techo-geeks say about political theory. I want to look at what they do. What are the values implicit in what they do—revealed-values, like revealed-preferences? This is the interesting feature of the Net's world.

And here is where Reagle's description gets to be very interesting. For there are two values that are central to the practice of the Internet's governance—values that link to the Open Source Movement and values that link as well to governance in real space. Let me describe these two and then develop with them some links to stuff we know about real space.

Open-Evolution

The first of these values we could call the value of *Open-Evolution*, and we could define it like this: Build a platform, or set of protocols, so that it can evolve in any number of ways; don't play god; don't hardwire any single path of development; don't build into it a middle that can meddle with its use. Keep the core simple and let the application (or end) develop the complexity.

Open-evolution flows out of the practice of Internet governance in part because the governors—in the terms of political philosophy—are liberals. They are neutral about the ends to which anyone might put the Net; they may not personally be neutral, but they have a modus vivendi that the Net not tilt in one way or the other.

But to preserve this neutrality, there are principles the system must preserve. One is an ideal of network design, first described at MIT, called "end-to-end."[24] The basic intuition is that the conduit be general; complexity should be at the end of the project. As Jerome Saltzer describes it, "don't force any service, feature, or restriction on the customer; his application knows best what features it needs, and whether or not to provide those features itself."[25]

The consequence of this end-to-end design is that the network is

24. *See* J.H. Saltzer et al., *End-to-End Arguments in System Design* <http://web.mit.edu/Saltzer/www/publications/endtoend/endtoend.pdf>.

25. Jerome H. Saltzer, *"Open Access" Is Just the Tip of the Iceberg* (Oct. 22, 1999) <http://mit.edu/Saltzer/www/publications/openaccess.html>.

neutral about how the network gets used. So long as an application follows the Internet protocols, it will run on the Net. Innovators need not get permission to run an application; designers need not worry that the owner of the Net will turn against them. The Net is architected to be neutral about its use, which means it can evolve however the users demand.

End-to-end is a constituting principle of the Internet. In my view, it is among the most important explanations for the Internet's extraordinary growth. But it is a second idea that I want to focus on here—an idea that also helps explain the freedom of the Net to evolve.

This is the ideal of modularity. Code can be written in any number of ways; good code can be written in only one way. Good code is code that is modular, and that reveals its functions and parameters transparently.

Why? The virtue is not efficiency. It is not that only good code will run. Rather, transparent modularity permits code to be modified; it permits one part to be substituted for another. The code then is open; the code is modular; chunks could be removed and substituted for something else; many forks, or ways that the code could develop, are possible. No one dictates which way the code will develop—this is what the anarchy means in Reagle's description. No rules say which way is right. Instead, the evolution of a market does that. The evolution of thousands of people trying their hand at improving a code, and thousands of people choosing which improvement makes sense.

The consequences both of these elements of what I am calling the open-evolution design and of that design itself is that no *one* can control how the system will evolve. No single individual gets to set the path the system will follow. It might evolve to follow a path, but it will evolve by the collective choice of many. Design of the Net then is the product of this bottom-up practice.

* * *

Here then we can make our first link. For consider the connection between this ideal of open-evolution and an aspect of antitrust's law of tying.

Tying doctrine says that in some cases, a seller can't bundle the

sale of one product with the sale of another.[26] Instead, the seller must offer the two separately, even if it offers the two together as well. *When* the law so insists is a complicated question—only sellers with market power in the tying product get scrutinized, and only if there is a threat of market power in the tied products. But for those few sellers who fall within this rule, the rule has a very important effect: like open-evolution, it says that the evolution of a particular product, or market, or set of products, can only be set by the decisions of a market. If the market chooses the bundled products over the separate products, all the better for the bundled products. But this is a decision for the market, not for the seller.

The law of tying functions, then, as a type of jurisdictional rule—when the threat of control or forcing is great, the law will assure that the market decides. And in this, it is just like open-evolution, for open-evolution too is a principle designed to assure that a market—and here, the market of developers pushing the Web in one way or the others—decides how the Web will develop. Not a king, not a president, and not even voting, but a consensus, and running code.

There is a second link as well, between this value of open-evolution and democracy. One value of democracy is that the democratic system should leave open the right of the democracy at least to develop in any way it chooses. The government is not to entrench one path or political party over another; it is not to tie support for it to other basic governmental services. It can't stack the electoral districts so that only one party wins;[27] it can't require that you vote Democratic in order to receive basic health care. It can't use its power in one area to tilt support to it elsewhere. Electors are to be free to pursue whatever path they want; the government is to leave open any path the electors want.

The point again is jurisdictional. Democracy is about preserving the jurisdiction of the people to choose the direction a government should go, just as the law of antitrust places in the market the jurisdiction to choose the direction that a product will go, just as open coders on the Internet place in the citizens of the Net the jurisdiction to choose the direction of that code. In all three contexts, the fundamental choice is a choice about jurisdiction; all three reserve ultimate power to a decentralized, bottom-up method of evolution.

26. *See* Jefferson Parish Hosp. v. Hyde, 466 U.S. 2 (1984).

27. *See, e.g.*, Davis v. Bandemer, 478 U.S. 109 (1986) (permitting constitutional challenge to districting based on political parties illegitimately entrenching power).

Universal Standing

Open-evolution is a first value of the Net—that control be from the bottom up, and that all paths be left open to bottom-up control. The second value we could call universal standing.

Go back to the evolution of GNU/Linux. GNU/Linux is an operating system; GNU/Linux is a pile of code. Anyone can download a copy of GNU/Linux to run as an OS; anyone can download a copy of GNU/Linux to get access to the code. This is the core of the Open Source Movement: that the code of open source software remains free for others to take, and modify, and use: that it sits in the public domain, which means that no one needs the permission of anyone else to take it, and improve it.

Without anyone's permission, one can improve it. Without going to work for any single company, one can work on it. One can improve it without getting the permission of some governmental bureaucrat. One can improve it without getting the authority of an election. Open code exists in the public domain for anyone to pick up just as Wiles' flawed proof existed in the public domain for anyone to fix. They are both common problems; they are both problems that anyone has standing to try her hand at solving.

Now your response here may well be:

> Yea, wonderful. We are all equally free to prove Fermat's Last Theorem; we are equally free to build out Linus' OS; and the poor and rich are equally free to sleep on the streets. In all three cases, there is a formal equality; in all three cases, one might ask, what value does this equality preserve?

The answer has nothing to do with equality, indeed, quite the opposite. Universal standing produces inequality, but its virtue is the way it produces inequality. For the world of Internet governance—whether the governance of bodies like the IETF, or the governance of the evolution of code like GNU/Linux—is not the world of the French Revolution. All citizens are not equally empowered. All citizens are not equally kings.

Recall: "We reject kings, presidents and voting. We believe in rough consensus and running code."

Now the unifying feature of "kings, presidents and voting" is formal power. To reject "kings, presidents and voting" is to reject formal power. Thus the Net rejects formal power. It rejects the idea that someone can say, "This is how the world will be," and that the world will be like that *simply because this is what was said.*

But this is not to say the Net rejects authority. Indeed, Clark's statement identifies the source of authority—"rough consensus and running code"—and between the two, I suggest, it is the second that is more important.[28] Universal standing keeps open the space for individuals to gain, not power, but authority. And individuals gain authority by their works. As Reagle (and someone else) says, "You will know them by their works." One gains authority not by a structure that says, "You are the sovereign," but by a community that comes to recognize who can write code that works.

This is the world of GNU/Linux as well. GNU/Linux is out there for anyone to take. But it does not evolve wholly on its own. There are choices about how it will evolve, and these choices get made by a council of elders—not old people, but people who have demonstrated that they can produce something. Anyone can propose a direction for the code, but proposals are not in English; proposals are in code. If there is a bug you think should be fixed in a particular way, you fix it, and offer the code that fixes it. It is the craft of your work and the elegance and power of your solution that commends it and gives it power. Not your status, not your rank, not your corporate position, not your friendships, but your code: running code, that by its power produces rough consensus.[29]

Now this more-complicated idea of universal standing, like the idea of open-evolution, also has some links to real-space ideals.

First it has links to the market: it remains our ideal, though governments have done much to screw this up, that markets are to remain open. Here again is a value that sings in the law of antitrust—that everyone formally be permitted to enter; that no one be shut out. You and I equally are assured an equal right to open a competitor to U.S. Steel, or Starbucks. The market guarantees open entry, which simply means it guarantees that everyone may enter the competition. But to guarantee equal entry is not to guarantee equal results.

28. Though Scott Bradner insists the first is important as well. It is true that the process doesn't get stopped by the veto of any minority, except if the process can't produce its idea in code. This difference in the design of the Internet's standards body, IETF, is discussed in Scott Bradner, *The Internet Engineering Task Force*, *in* OPEN SOURCES—VOICES FROM THE OPEN SOURCE REVOLUTION, *supra* note 14, at 47. Other standards bodies—for example, IEEE—allow standards to be adopted that have not been shown to work.

29. There is a related ethic in the world of MUDs—text based virtual realities. Power in these spaces is often allocated to characters called "Wizards." The ethic is that wizards should be people who can code well, and when the power gets allocated by favoritism, this is not viewed favorably by others within the MUD. *See* Elizabeth Reid, *Hierarchy and Power: Social Control in Cyberspace*, *in* COMMUNITIES IN CYBERSPACE 120 (Marc. A. Smith & Peter Kollack eds., 1999).

Winners are to be chosen by the market's choice; the market too rejects kings, presidents, and voting.

And so too does it have links to democracy—or to ideals of democracy. Whether realized or not, the ideal of democracy is that anyone can enter—regardless of class, regardless of blood, anyone is permitted to put their ideas up for selection. This is our ideal, even though we realize that in real space only crazies and the rich really ever put themselves up for election. And even though in real space we understand that those who prevail rarely prevail for the right reason. In cyberspace, this ideal seems real. Vint Cerf, David Clark, Jon Postel, John Gilmore, Richard Stallman, Linus Torvalds—these were nobodies who became somebodies for a reason, just as we imagine that Lincoln, and Webster, and Wilson (but who else?) were nobodies who became somebodies for a reason. The value is that anyone can stand; that democracy, like the market, like open code, leaves the competition open.

* * *

I've identified two values that can be considered central to the Internet's governance: that it be designed to evolve however we choose; that it remain open to anyone's work. Two values that are central to the way the market works: that it be designed to evolve however we choose; that it remains open to entry by anyone. Two values that are central to democracy: that government be designed to evolve however we choose; that it remain open to anyone's work. Two values that link three critical domains of our life in cyberspace. Do they say something then about the alternative—about code that is not open, about code that is not free? Do these two values entail something conclusive about company code?

"Conclusive" is a hard word for an academic, but let's start with this. Nothing entails that open and closed can't coexist. Our world has commons; it has private property as well. There is science in universities; there is science that is proprietary. The lesson is not an extreme; the key is balance. Balance entails that one side does not insist on the death of the other; balance entails the survival of the other.

Linus Torvalds put it better than I could: When the question was raised whether proprietary code could be tied to GNU/Linux, and many in the Open Source Movement had opposed it, Torvalds said, "My opinion on licenses is that 'he who writes the code gets to choose

his license, and nobody else gets to complain.' Anybody complaining about a copyright license is a whiner."

Torvalds is no fanatic. He is no Free Software extremist.[30] He imagines a balance between commons and private. That just as we live in real space with common and private properties, so too can cyberspace live with common and private properties.

But the point of the links that I have tried to suggest is that the values that are at stake in this trade-off are more than the values of efficiency. The values of Internet governance are about the values of governance; and the balance the Internet must strike is between that part that leaves itself open to these values, and that part that doesn't.

There's a part that must lie in the commons. There's a part that, like basic science and democratic politics, must remain open to all. That just as we don't privatize every public park, every street, and every idea, we can't privatize every feature of cyberspace. Can't privatize, which is not to say that government must control it. The opposite of private is not the government. The opposite of private is the commons. It is the place where public streets are, the place where Fermat's Last Theorem lay, the place where everyone remains free to push how things are to something different. There is a value in preserving that space, regardless of efficiency.

The values of a market, the values of democracy, the values of our tradition of Internet governance—all these show a place for the commons; and open systems and open code are the Internet's commons. Not an exclusive role, but a central role. And the challenge for our future is to understand this balance in a world where the network becomes the OS.

30. Neither is the Free Software Foundation founder, Stallman. Stallman too believes there is a space for both sorts of license. The ideal proportions, however, may well differ between Torvalds and Stallman. *See* Stallman, *supra* note 15, at 63, 66-70.