CODE IS SPEECH: Legal Tinkering, Expertise, and Protest among Free and Open Source Software Developers

GABRIELLA COLEMAN
Department of Media, Culture, and Communication, NYU

We do not act because we know. We know because we are called upon to act.

—Johann Gottlieb Fichte

THE POETICS OF LAW, TECHNOLOGY, AND PROTEST

Like many computer aficionados today, Seth Schoen writes all of his software as "Free and Open Source Software" (F/OSS) to ensure that the source code—the underlying directions of computer programs—will remain accessible for other developers to use, modify, and redistribute. In so doing, a Free Software developer like Schoen not only makes technology but also participates in an emerging effort that redefines the meaning of liberal freedom, property, and software by asserting in new ways that code is speech.

For example, a tiny portion of a 456 stanza haiku written by Schoen in February 2001 makes just this claim:

Programmers' art as
that of natural scientists
is to be precise,

complete in every
detail of description, not
leaving things to chance.

Reader, see how yet
technical communicants
deserve free speech rights;

see how numbers, rules,
patterns, languages you don't
yourself speak yet,

still should in law be
protected from suppression,
called valuable speech!

Schoen's protest poem not only argued that source code is speech but also demonstrated it: the extensive haiku was in fact a transcoding of a short piece of Free Software called DeCSS, which could be used to decrypt access controls on DVDs in violation of current copyright laws. Schoen wrote this poem not only to be clever but also as part of a worldwide wave of protests following the arrest of DeCSS's coauthor, Jon Lech Johansen, and the lawsuits launched against those who published DeCSS.

Twenty years ago there was no general understanding connecting code and speech. Today Schoen and hundreds of other developers routinely make just this assertion and the resulting association between free speech and source code is now one of the most frequently used fixtures among F/OSS developers, who have become extremely well versed in law, politically aware, and committed to free speech issues, as well as technically adept at promoting their cause. How did these surprising transformations come about?

In this essay, I examine the ways in which F/OSS developers like Schoen are reconfiguring what source code and speech mean ethically, legally, and culturally, and the broader political consequences of these redefinitions. I demonstrate how developers refashion liberal precepts in two distinct cultural "locations" (Gupta and Ferguson 1997), the F/OSS project and broader legal battles, resulting in what legal theorist Robert Cover calls "jurisgenesis": the collective construction of new legal meanings and artifacts that diverge from statist or dominant interpretations of the law (1992). Developers construct new legal meanings by challenging the idea of software as property and by crafting new free speech theories to defend the idea of software as speech.

Using a framework of jurisgenesis, this essay does three things. First, it demonstrates how F/OSS developers explore, contest, and specify the meaning of liberal freedom—especially free speech—via the development of new legal tools and discourses within the context of the F/OSS project. In particular, I highlight how developers concurrently tinker with technology and the law using similar skills, which transform and consolidate ethical precepts among developers. Using Debian, the largest Free Software project in the world, as my primary ethnographic example, I suggest that these F/OSS projects have served like an informal law education, transforming technologists into informal legal scholars who are experts in the legal technicalities of F/OSS as well as proficient in the current workings of intellectual property law.[1]

Second, I examine how these developers marshal and bolster this legal expertise during broader legal battles to engage in what Tilly and Tarrow describe as "contentious politics" (2006). I concentrate on the separate arrests of two programmers, Jon Johansen and Dmitry Sklyarov, and on the protests, which unfolded between 1999 and 2003, that they provoked. These critical events led to a historically unprecedented proliferation of claims connecting source code to speech, with Schoen's 456-stanza poem providing one of many well-known examples. These events are historically notable because they dramatized what normally exists more tacitly and brought visibility to two important social processes. They publicized the direct challenge that F/OSS represents to the dominant regime of intellectual property (and thus clarified the democratic stakes involved) and also made more visible, and thus stabilized, a rival liberal legal regime intimately connecting source code to speech.

Third, the story I relate here helps deepen our understanding of why, and especially when F/OSS, a technology based movement, emerged in such politicized ways and historicize what Kelty (2008) calls a recursive public—a public constituted by participants who defend the right to make and alter technology through argument and by tinkering with the technologies (notably the Internet) through which they collectively associate. It was during this period, between 1999 and 2003, when hackers were arrested and new intellectual property instruments ominously loomed, that participants of this public worked out, specified, and clarified the political significance of their technical work.

In what follows, I begin by introducing the basic history of Free Software and presenting some general attributes concerning legal tinkering that cut across a number of F/OSS projects. I then examine legal production within one specific project: Debian. The final section covers the historical moment when F/OSS's particular public stance for free speech was rearticulated anew for wider public consumption. I conclude by briefly considering the importance of legal expertise among activists participating in global politics today.

THE BIRTH OF THE LEGAL HACKER

F/OSS is produced by tens of thousands of technologists located around the globe who work together on projects, such as the Web browser Firefox, over the Internet. In the span of a decade these projects, which were loosely organized and decentralized much like a virtual "bazaar" (Raymond 1999), have grown into large semiformal institutions with complex governance structures. Although many projects still deploy informal and ad hoc decision making for development, most mid to large sized now rely on formal mechanisms—voting protocols, constitutions, codes of conduct, legal exams, and detailed policy requirements—to coordinate technical production (Kelty 2008; O'Mahony and Ferraro 2007; Weber 2004). Many of the participants—some volunteers, some paid—also refer to themselves with pride as hackers, that is, as computer aficionados driven by an inquisitive passion for tinkering and committed to an ethical version of information freedom (Castells 2001; Coleman and Golub 2008; Himanen 2001; Thomas 2002; Wark 2004). Increasingly, hackers' ethical stances, I suggest, are expressed through legal idioms, which are now pervasive in the arena of F/OSS. However, the law was initially not only absent among hackers but also patents and copyrights seemed to be the primary obstacle, funneling software into a proprietary model.

It is now well known that Free Software first emerged as the brainchild of MIT hacker Richard Stallman, who in 1985 founded the Free Software Foundation and began writing what he called Free Software—software that, unlike proprietary software, could be copied, shared, circulated, and modified. What is less known is that when initiating this fight, Stallman did not conceptualize Free Software in legal terms. His goal was not to tinker with the law, which he knew very little about, but to write a suite of Free Software applications to replace proprietary software, thus circumventing the problem of the law.

As it turned out, the law was not entirely a foe (in the form of copyright and patents); it became indeed a friend, at least once Stallman reconfigured it properly. Because of a fairly complicated controversy (whose details need not concern us here but in which Stallman was accused of illegally copying source code) the legal issues concerning patents, copyrights, and public domain first and palpably became clear to software developers (for the detailed history, see Kelty 2008:ch. 6). By 1989, Stallman had crafted a legal framework for Free Software to prevent the type of controversy that had erupted over his first Free Software program from recurring, and to add a layer of transparency, control, and accountability for Free Software.

Stallman's legal solution, the General Public License (GPL), commonly referred to as the copyleft, uses copyright law, a Constitutional mandate, to undermine the logic of copyright law. The GPL is built on copyright, but disables the restrictions of copyright to allow for modification, distribution, and access; it is also self-perpetuating because it requires others to adopt the same license if they modify copylefted software. By inventing the copyleft, Stallman provided the rudiments of a rival liberal legal vocabulary of freedom, which hackers would eventually appropriate and transform to include a more specific language of free speech.

The first developers and hackers associated with Free Software were for the most part users, not producers, and possessed only a rudimentary understanding of legal issues. Some were even repelled by Stallman's vision of software freedom, which he had outlined in 1985 in his GNU Manifesto—a public declaration of the ethical principles and intentions behind Free Software. One developer explained his ambivalence during an interview in 2002 as follows: "I was a little confused. To me it [the GNU Manifesto] sounded socialistic and ideological, a bit like [the] Jehovah's Witness, something which will never come to pass. At the time I disregarded it as a mad man's dream. But I did continue to use [the GNU Free Software] Emacs and GCC." This developer, as well as many developers I have interviewed, is now firmly committed to some of the ethical and pragmatic principles of Free Software, and has expert command not only of the technology but also of its legal underpinnings.

It would take the Linux kernel project to transform Free Software from a "mad man's dream" into a large-scale movement with thousands of contributors, many of whom would eventually commit to—and alter—the legal and ethical principles of Free Software. Linus Torvalds, who wanted to rewrite the proprietary UNIX operating system for the personal computer, initiated the Linux kernel project in 1991 as a hobbyist pursuit and eventually used an electronic mailing list to request feedback. Within the year, volunteer programmers were turning Linux into a high-quality PC operating system. Because coordination was organized virtually and informally, programmers greeted Linux with bewilderment—but with open arms. The experience of discovering that there existed an (almost) fully working UNIX system for a PC, with available source code was, as put by one developer during an interview, "jaw dropping." It was surprising how hackers, working informally over the Internet, could produce a reliable piece of software. This was a period when Free Software development resembled a "great babbling bazaar of differing agendas and approaches out of which a coherent and stable system could seemingly emerge only by a succession of miracles" (Raymond 1999).

Most developers did not wax philosophical in search of the success of this new way of working. Instead they continued its success by initiating similar efforts. For example, in 1993 Ian Murdock, a computer science student, combined Torvald's kernel with some of the GNU/FSF software tools to create a fully functional operating system distribution of Linux called Debian. Scores of other projects, such as the Apache Web server and the graphical user interface GNOME, were also initiated at this time.

As these virtual organizations got off the ground in the mid- to late 1990s, Eric Raymond, a libertarian-leaning hacker, sought to refashion the public persona–presentation of Free Software to attract business investors (Raymond 1999). To do so, he replaced the term "Free Software" with the ostensibly nonideological terminology of "Open Source Software." Although Free Software foremost emphasizes the right to learn and to access knowledge, "Open Source" flags practical benefits of what are the same collaborative methods, licenses, and virtual organizations (Kelty 2008). Despite these differences, it is difficult to identify many purists on either side; participants often appeal to both moral and utilitarian logics, although they might emphasize one logic over the other.

More important for our story of the law was the fact that during this period F/OSS projects matured into semiformal institutions. Largely to manage the influx of volunteers, most midsized to large projects supplemented, although they did not displace, the bazaar style of work with more formal mechanisms (voting protocols, membership and policy procedures, codes of conduct) to aid in social and technical coordination. It is crucial to note here that these institutions also became de facto training grounds, where developers acquired new ethical and legal skills, idioms, and viewpoints.

The growing presence of the law in F/OSS projects was an outgrowth of three circumstances. First, to participate effectively in technological production, developers have had to acquire basic legal knowledge to make certain informed decisions. For example, when licensing their own software, they must choose one of over five dozen F/OSS licenses, or ascertain whether the software license on the software package they maintain is compliant with existing license standards and guidelines. Second, developers produce their own legal artifacts, such as licenses, charters, and legal tests, and as a result there is a tremendous body of legal exegesis in the everyday life of their F/OSS project. Third, many developers closely track news about Free Software–related legal battles, especially those seen to impinge on their productive freedom: Has the patent directive passed in the European Union Parliament? How has Diebold, the voting machine company, used the Digital

Millennium Copyright Act, to suppress the circulation of information incriminating the company? Information regarding these and other relevant developments is not only posted widely on the Internet but also many developers get directly involved in these cases, as the reader will see in the next section.

To be sure, there are some developers and hackers who express distaste for discussions of legal policy and actively distance themselves from this domain of "polluting politics." But even though hackers will assert the superiority of technical to legal language, or even technical to legal labor (some hackers claim that the law is a waste of time; or as stated a bit more cynically by one developer, "writing an algorithm in legalese should be punished with death . . . a horrible one, by preference"), technologists can very quickly acquire legal fluency and literacy.

One reason for this facility, I suggest, is that the skills, mental dispositions, and forms of reasoning necessary to read and analyze a formal, rule-based system like the law parallel the operations necessary to code software. Both, for example, are logic-oriented, internally consistent textual practices that require great attention to detail. Small mistakes in both law and software—a missing comma in a contract or a missing semicolon in code—can jeopardize the integrity of the system and compromise the intention of the author of the text. Both lawyers and programmers develop mental habits for making, reading, and parsing what are primarily utilitarian texts. As noted by two lawyers who work on software and law, "Coders are people who write in subtle, rule-oriented, specialized, and remarkably complicated dialects," which, they argue, pertains also to how lawyers make and interpret the law (Cohn and Grimmelmann 2003).[2]

This helps us understand why it's been so easy for developers to integrate the law into everyday technical practice and advocacies, and avoid some of the frustration that afflicts lay advocates trying to acquire legal fluency to make larger political claims. Kim Fortun, for example, describing the activists who worked on behalf of the victims of the Bhopal disaster, perceptively shows how acquiring legal fluency and developing the correct legal strategy is frustrating and can lead to cynicism (2001:ch. 1). Many hackers are similarly openly cynical about the law because it is seen as easily subject to political manipulation. Despite this cynicism, I never encountered any expression of frustration about the actual process of learning the law. A number of developers I worked with clearly enjoy learning and arguing about a pragmatic subset of law (such as a particular legal doctrinal framework), just as they do technology. Many developers apply the same skills required for hacking to the law and, as we will see, technology and the law at times seamlessly blend into each other.

To give a taste of this informal legal scholarship—of the relationship between technical expertise and legal understanding, and of the ways in which legal questions are often tied to moral issues—in one Free Software project, I will describe some of Debian's legal micropractices: its routine legal training, advocacy, and exegetical legal commentary. These legal micropractices allow for what Robert Cover, in his description of jurisgenesis, calls a "commitment in living out legal meaning" (Cover 1992:103). For new legal precepts to become meaningful, Cover insists, they must be incorporated into the practice of everyday life through a process that stretches from informal narrative to formal exegesis of existing precepts. This occurs in many F/OSS projects, but the process is especially salient in Debian. To deepen this picture of how developers live in and through the law, I proceed to a broader struggle, one where similar legal processes are underway, but are more visible because of the way they have circulated beyond the boundaries of projects proper.

"LIVING OUT LEGAL MEANING" WITHIN A FREE SOFTWARE PROJECT

Just over a thousand volunteers are participating in the Debian project at this time (early 2009), writing and distributing a Linux-based operating system composed over 20 thousand individual software applications. In its nascence, Debian was run entirely informally; it had fewer than two dozen volunteers who communicated primarily through a single e-mail list. To accommodate growth, however, significant changes in policy, procedures, and structure occurred between 1997 and 1999. Now Debian boasts a complex hybrid political system; a developer Internet relay chat (IRC) channel; a formalized membership entry procedure called New Maintainer (NM); and a set of charters that includes a Constitution, a Social Contract, and the Debian Free Software Guidelines (DFSG).

A few Debian developers are paid by their employers to work on Debian, but the project itself pays no one, not even the release managers, some of whom spend 40 hours or more a week for two to three months on the final release of a new version of Debian. Generally, developers attain positions of authority through a meritocratic system that rewards those who perform exceptional work and who wish to occupy a position of responsibility (although there are exceptions). Each developer decides where and how to contribute, with no formal mandates from those with organizational authority to direct developer labor.

Despite this hands-off, self-directed approach all developers go through an initial vetting process—the New Maintainer (NM) process. It is an obligatory point of passage (Latour 1988) through which prospective developers apply for membership. Fulfilling the mandates of the NM can take months, even years, of hard work: a prospective developer has to find a sponsor and advocate; learn the in and outs of Debian policy, Free Software licensing, and technical infrastructure; successfully package software that satisfies a set of technical standards; meet at least one other Debian developer in person for identity verification; and pass a series of written tests on technical, philosophical, and legal matters. Thus do prospective developers become familiar with the active legal culture of Debian.

Several questions in the NM application cover what is now one of the most famous philosophical and legal distinctions in the world of Free Software: "free beer" versus "free speech." Common among developers today, this distinction arose only recently, during the early to mid-1990s.

A prospective Debian Developer (DD) describes the difference in an NM application:

> Free speech is the possibility of saying whatever one want wants to. Software [that is] free as in beer can be downloaded and used for free, but no more. Software [that is] free as in speech can be fixed, improved, changed, be used as building block for another [*sic*] software.

Some developers also note that their understanding of "free speech" is nested within a broader liberal meaning codified in the constitutions of most liberal democracies:

> Used in this context the difference is this: "free speech" represents the freedom to use/modify/distribute the software as if the source code were actual speech which is protected by law in the US by the First Amendment . . . "free beer" represents something that is without monetary cost.

This differentiation between free beer and free speech is the clearest enunciation of what, to these developers, are the core meanings of "free"—expression, learning, and modification. Freedom is understood foremost to be about personal control and autonomous production and decidedly not about commodity consumption or "possessive individualism" (Macpherson 1962), a message that is constantly restated by developers: Free Software is "free as in speech, not beer."

This distinction may seem simple, however, the licensing implications of "freedom" and free speech are complicated enough that the NM process continues with a series of very technically oriented questions whose answers start to enter the realm of legal interpretation. Many of these concern the Debian Free Software Guidelines (DFSG), a set of ten provisions by which to measure whether a license can be considered "free." Of these questions, one or two are fairly straightforward. For example:

> "Do you know what's wrong with Pine's current license in regard to the DFSG?"
> After looking at the license on the upstream site it is very clear why pine is non-free. It violates the following clauses of the DFSG:
>
> 1.) No Discrimination Against Fields of Endeavor—it has different requirements for non-profit vs. profit concerns.
> 2.) License Must Not Contaminate Other Software—it insists that all other programs on a CDROM must be "free-of-charge, shareware, or non-proprietary"
> 3.) Source Code—it potentially restricts binary distribution [binary refers to compiled source code]

The sample license for an e-mail program, Pine, violates a number of DFSG provisions. With different provisions for nonprofit and for-profit endeavors, as an example, it discriminates according to what the DFSG calls "fields of endeavor."

Developers are then asked a handful of far more technical licensing questions, among them:

> At http://people.debian.org/~joerg/bad.licenses.tar.bz2 you can find a tarball of bad licenses. Please compare the graphviz and three other (your choice) licenses with the first nine points of the DFSG and show what changes would be needed to make them DFSG-free.

The answer clearly demonstrates the depth of legal expertise required to address these questions: "Remove the discriminatory clauses . . . allow distribution of compiled versions of the original source code . . . replace [sections] 4.3 with 4.3.a and 4.3.b and the option to choose . . . "

After successfully finishing the NM process, some developers think only rarely about the law or the DFSG, perhaps only tracking legal developments of personal interest. Even if a developer is not actively learning the law, however, legal discourse is nearly unavoidable because of the frequency with which it appears on Debian mailing lists or chat channels. Informal legal pedagogy thus continues long after the NM.

As an example, I quote an excerpt from a discussion on IRC wherein a developer proposed a new Debian policy that would clarify how non–Free Software packages (those noncompliant with their license guidelines) should be categorized so as to make it absolutely clear how and why they cannot be included in the main software repository, which can only have Free Software.

> <dangmang> Markel: what is your opinion about making a recommendation in policy that packages in non-free indicate why they're in non-free, and what general class of restrictions the license has?
> <Markel> dangmang: well, I am not too keen on mandating people do more work for non-free packages. but it may be a good practice suggestion
> <JabberWalkie> dangmang: Then I would suggest that the ideal approach would be to enumerate all the categories you want to handle first, giving requirements to be in those categories.
> <dangmang> Markel: true. could the proposal be worded so that new uploads would have to have it?
>
> [. . .]
>
> <JabberWalkie> dangmang: You don't want to list what issues they fail; you want to list what criteria they meet.
>
> [. . .]
>
> <JabberWalkie> dangmang: X-Nonfree-Permits: autobuildable, modifiable, portable
> <Markel> the developers-reference should mention it, and policy can recommend it, for starters
> <Markel> dangmang: we need to have well defined tags
> <JabberWalkie> mt3t: "gfdl", "firmware".
>
> [. . .]
>
> <JabberWalkie> mt3t: No "You may not port this to _____".
> <JabberWalkie> mt3t: You wouldn't believe what people put in their licenses. :)
> <dangmang> Markel: right . . . I think I'll start on the general outline of the proposal, and flesh things out, and hopefully people will have comments to make in -policy too when I start the procedure.

It is not the exact legal or technical details that I mean to emphasize, but how, late on a Friday night (when the discussion happened), a developer made a policy recommendation and his peers immediately offered advice on how to proceed, discussing the issue with such sophisticated legal vocabulary that to the uninitiated it appears completely obscure. This is simply part of the "natural" social landscape of most Free Software projects.

More formal legal avenues are employed, however. Debian developers may contact the original author (called the upstream maintainer) of a piece of software that they are considering including and maintaining in Debian. Many of these exchanges concern licensing problems that would keep the software out of Debian; in this way, non-Debian developers also undergo informal legal training. Sometimes developers act in the capacity of legal advocates, convincing these upstream maintainers to switch to a DFSG-compliant license, which is necessary if the software is to be included in Debian.

The developers who hold Debian-wide responsibilities must in general be well versed in the subtleties of F/OSS licensing. The ftpmasters, whose job is to integrate new software packages into the main repository, must check every single package license for DFSG compatibility. Distributing a package illegally could leave Debian open to lawsuits.

One class of Debian developers has made legal matters their obsession. These aficionados contribute prolifically to the legal pulse of Debian in debian-legal—a mailing list that, because of its large number of posts, is not for the faint of heart. For those who are interested in keeping abreast but don't have time to read every message posted on debian-legal, summaries link to it in a weekly newsletter, "Debian Weekly News." Below, I quote a fraction (about one-fifth) of the legal news items that were reported in DWN during the course of 2002 (the numbers are references linking to mailing list threads or news stories):

> **GNU FDL a non-free License?** Several [22] people are [23] discussing whether the [24]GNU Free Documentation License (GFDL) is a free license or not. If the GFDL is indeed considered a non-free license, this would [25] render almost all KDE and many other well known packages non-free since they use the GNU FDL for the documentation. Additionally, here's an old [26] thread from debian-legal, which may shed some light on the issue.
>
> **RFC: LaTeX Public Project License.** Claire Connelly [4] reported that the LaTeX Project is in the process of considering changes to the LaTeX Project Public License. She tried to summarize some of the concerns that Debian people have expressed regarding the changes. Hence, Frank Mittelbach asked for reviews of the draft of version 1.3 of the [5]LaTeX Public Project License rather than of the current version (1.2).
>
> **Enforcing Software Licenses.** Lawrence Rosen, general counsel for the [20]Open Source Initiative, wrote an [21]article about the enforceability of

> software licenses. In particular, he discusses the issue of proving that somebody assented to be bound by the terms of a contract so that those terms will be enforced by a court. Authors who wish to be able to enforce license terms against users of their source code or compiled programs may find this interesting.
>
> **Problematic BitKeeper License.** Branden Robinson [3]pointed out that some of us may be exposed to tort claims from BitMover, Inc., the company that produces BitKeeper, the software that is the primary source management tool for the Linux kernel. Your license to use BitKeeper free of charge is revoked if you or your employer develop, produce, sell, or resell a source management tool. Debian distributes rcs, cvs, subversion and arch at least and this seems to be a [4]different case. Ben Collins however, who works on both the Linux kernel and the subversion project, got his license to use BitKeeper free of charge [5]revoked . . .

These are newsletter summaries, which are read by thousands of developers outside of the Debian community proper as well as by Debian developers. It is also worth noting how outsiders turn to Debian developers for legal advice and how legal expertise is valued. Practical and immediate concerns are layered upon global currents and more philosophical musings. Some discussions can be short, breeding less than a dozen posts; other topics are multiyear, multilist, and may involve other organizations, such as the Free Software Foundation. These conversations may eventually expand and reformulate licensing applications.

One routine task undertaken in debian-legal is to help developers and users choose appropriate licensing, by providing in-depth summaries of alternative licenses compliant with the DFSG. One such endeavor was to determine whether a class of Creative Commons (CC) licenses (developed to provide creative producers, such as musicians and writers, with alternatives to copyright) was appropriate for software documentation. Debian developers assessed that the CC licenses under consideration failed to meet the standards of the DFSG, and suggested that Debian developers not look to them as licensing models. The most remarkable aspect of their analysis is that it concludes with a detailed set of recommendations for alterations to make the CC licenses more “free” according to the Debian licensing guidelines. In response to these recommendations, Lawrence Lessig of Creative Commons contacted Evan Prodromou, one of the authors of this analysis, to try to find solutions to the incompatibilities between the DFSG and some of the CC licenses.

There is something ironic, on the one hand, about a world-renowned lawyer contacting a bunch of geeks with no formal legal training to discuss changes to the licenses that he created. On the other hand, who else would Lessig contact? These developers are precisely the ones inhabiting this legal world. These geeks are training themselves to become legal experts, and much of this training occurs in the institution of the Free Software project.

Debian's legal affairs don't just produce what a group of legal theorists have identified as everyday legal awareness (Ewick and Silbey 1998; Mezey 2001; Yngvesson 1989). The arena of F/OSS probably represents the largest single association of amateur intellectual property and free speech legal scholars ever to have existed. Given the right circumstances, many developers will marshal this expertise as part of broader, contentious battles over intellectual property (IP) law and the legality of software, the topic of the next section.

CONTENTIOUS POLITICS AND THE STABILIZATION OF CODE AS SPEECH

If hackers acquire legal expertise by participating in F/OSS projects, they also use and fortify their expertise during broader legal battles. Here I examine one of the most heated of the recent controversies over intellectual property, software, and access: the arrests of Jon Johansen and Dmitry Sklyarov. These provoked a series of protests and produced a durable articulation of a free speech ethic that, under the umbrella of free and open source software development, had been under quieter cultivation in the previous decade. Intellectual property has been debated since its inception (Hesse 2002; Johns 2006; McGill 2002), but as media scholar Siva Vaidyanathan notes, these previous debates have "rarely punctured the membrane of public concern" (2006:298). It was precisely in this period (1999 to 2003) and in part because of these events, when a more visible, notable, and "contentious politics" (Tilly and Tarrow 1996) over IP emerged, especially in the United States and Europe.[3]

Before discussing how the emergence of a "contentious politics" worked to stabilize the connection between speech and code, some historical context is necessary. At the most general level, we can say a free speech idiom formed as a response to the excessive copyrighting and patenting of computer software. (Prior to 1976, this had been rare.) The first widely circulated paper associating free speech and source code was "Freedom of Speech in Software" (1991) written by a programmer, Peter Salin. He characterized computer programs as "writings" to argue that software was unfit for patents, although appropriate for copyrights and, thus, free-speech protections. The idea that coding was a variant of writing was gaining traction also, in part, because of the popular publications of Stanford Computer Science professor Donald Knuth on the art of programming (Black 2002; Knuth 1998). During the early 1990s, a new ethical sentiment emerged among early USENET enthusiasts (many of them hackers and developers), that the Internet should be a place for unencumbered free speech (Pfaffenberger 1996). This sensibility in later years would become specified and attached to technical artifacts such as source code Perhaps most significantly, what have come to be known as the "encryption wars" were in the mid-1990s waged over the right to freely publish and use software cryptography in the face of governmental restrictions that classified strong forms of encryption as munitions. The most notable juridical case in these struggles was *Bernstein v. U.S. Department of Justice.* The case opened in 1995, after a computer science student, Daniel J. Bernstein, sued the government to challenge international traffic in arms regulations (ITAR), which classified certain types of strong encryption as munitions and thus subjected them to export controls. Bernstein could not legally publish or export the source code of his encryption system, Snuffle, without registering as an arms dealer. After years in court, in 1999 the judge presiding over the case concluded that government regulations of cryptographic "software and related devices and technology are in violation of the First Amendment on the grounds of prior restraint."[4]

Neither Salin's article nor the Bernstein case questioned copyright as a barrier to speech. With the rise of Free Software, developers launched a far more extensive critique of copyright. The technical production of Free Software had trained developers to become legal thinkers and tinkerers well acquainted with the intricacies of IP law as they became committed to an alternative liberal legal system steeped in discourses of freedom and increasingly free speech. Thus, if the first free speech claims among programmers were proposed by a handful of developers and deliberated in a few court cases in the early to mid-1990s, in the subsequent decade they grew social roots in the institution of the F/OSS project; individual commitments and intellectual arguments grew into a full-fledged collective social practice anchored firmly in F/OSS technical production.

Unanticipated state and corporate interventions, however, raised the stakes and gave this rival legal morality a new public face. Indeed, it was only because of a series of protracted legal battles that the significance of hacker legal expertise and free speech claims became apparent to me. I had, like so many developers, not only taken their free speech arguments about code as self-evident but also had taken for granted their legal skills in the making of these claims. Witnessing and participating in the marches, candlelight vigils, street protests, and artistic protests (many of them articulated in legal terms), among a group of people who tend to shy away from such overt forms of traditional political action (Coleman 2004; Galloway 2004; Riemens 2003), led me to seriously reevaluate the deceptively simple claim: that code is speech. In other words, what existed tacitly became explicit after a set of exceptional arrests and lawsuits.[5]

POETICALLY PROTESTING THE DIGITAL MILLENNIUM COPYRIGHT ACT

On October 6, 1999, a 16-year-old Norwegian programmer, Jon Johansen, used a mailing list to release a short, simple software program, DeCSS. Written by Johansen and two anonymous developers, DeCSS unlocks the Digital Rights Management (DRM) on DVDs. Before DeCSS, only computers using either Microsoft's Windows or Apple's operating system could play DVDs; Johansen's program allowed Linux users to unlock a DVD's DRM to play movies on their computers. Released under a Free Software license, DeCSS soon was being downloaded from hundreds, possibly thousands, of Web sites. In the hacker public, the circulation of DeCSS would transform Johansen from an unknown geek into a famous "freedom fighter"; entertainment industry executives, however, would soon seek out his arrest.

Although many geeks were gleefully using this technology to bypass a form of Digital Rights Management so they could watch DVDs on their Linux machines, various trade associations sought to ban the software because it made it easier to copy and thus pirate DVDs.[6] In November 1999, soon after its initial spread, the DVD Copy Control Association and the Motion Picture Association of America (MPAA) sent cease-and-desist letters to more than fifty Web site owners and Internet service providers, requiring them to remove links to the DeCSS code for its alleged violation of trade secret and copyright law and, in the United States, the Digital Millennium Copyright Act (DMCA). Passed in 1998 to "modernize" copyright for the digital age, the DMCA's most controversial provision outlaws the manufacture and trafficking of technology (which can mean something immaterial, such as a six-line piece of source code, or something physical) capable of circumventing copy or access protection in copyrighted works that are in a digital format. The DMCA makes illegal the trafficking and circulation of such a tool, even if it can be used for lawful purposes (such as fair use copying) or is never used.[7]

In January 2000, the MPAA filed three lawsuits to stop DeCSS. One was against the well-known hacker organization and publication *2600* and its founder

Eric Corley (more commonly known by his hacker handle, Emmanuel Goldstein). He would fight the lawsuits, appealing to *2600*'s journalistic free-speech right to publish DeCSS. As happens with censored material, the DeCSS code at this time was unstoppable; it spread like wildfire.

Simultaneously the international arm of the MPAA urged prosecution of Johansen under Norwegian law (the DMCA, an American law, had no jurisdiction there). The Norwegian Economic and Environmental Crime Unit (ØKOKRIM) took the MPAA's informal legal advice and indicted Johansen on January 24, 2000, for violating an obscure Norwegian criminal code. Johansen (and his father) were arrested and released on the same day, and law enforcement confiscated his computers. He was scheduled to face trial three years later.

Hackers and other geek enthusiasts discussed, debated, and decried these events, and a few consistent themes and topics emerged. The influence of the court case discussed above, *Bernstein v. United States,* was one such theme. This case established that software could be protected under the First Amendment, and in 1999 caused the overturning of the ban on exportation of strong cryptography. Programmers could write and publish strong encryption on the grounds that software was speech.

F/OSS advocates, seeing the DeCSS cases as a similar situation, hoped that the courts just might declare DeCSS worthy of First Amendment protection. Consider the first message posted on the mailing list "dvd-discuss"—a mailing list that would soon attract a multitude of programmers, F/OSS developers, and activist lawyers to discuss every imaginable detail concerning the DeCSS cases:

> I see the DVD cases as the natural complement to Bernstein's case. Just as free speech protects the right to communicate results about encryption, so it protects the right to discuss the technicalities of decryption. In this case as well as Bernstein's, the government's policy is to promote insecurity to achieve security. This oxymoronic belief is deeply troubling, and worse endangers the very interests it seeks to protect.[8]

There were, it turned out, significant differences between Bernstein and DeCSS. In the Bernstein case, hackers were primarily engaged spectators. Furthermore, many Free Software advocates were critical of Bernstein's decision to copyright, and thus tightly control, all of his software. In the DeCSS and DVD cases, by contrast, many F/OSS hackers became participants, by injecting into the controversy notions of Free Software, free speech and source code (a language they were already fluent in from F/OSS technical development). Hackers saw

Johansen's indictment and the lawsuits not simply as a violation of their right to software, but their right to produce F/OSS. As the following call-to-arms reveals, many hackers understood the attempt to restrict DeCSS as a "full-fledged war against the Open Source movement":

> . . . here's why they're doing it: **Scare tactic** . . . I know a lot of us aren't political enough—but consider donating a few bucks and also mirroring the source. . . . This is a full-fledged war now against the Open Source movement: they're trying to stop . . . everything. They can justify and rationalize all they want—but it's really about them trying to gain/maintain their monopoly on distribution . . . [9]

Johansen was, for hackers, the target of a law that challenged, fundamentally, their freedom to tinker and to write code—values that acquired coherence and had been articulated in the world of F/OSS production.

Hackers moved to organize politically. Many websites providing highly detailed information about the DMCA, DeCSS, and copyright history went live, and the Electronic Frontier Foundation (EFF) launched a formal "Free Jon Johansen" campaign. All this was helping to stabilize the growing links between source code and software, largely because of the forceful arguments that computer code is expressive speech. Particularly prominent was an amicus curiae brief on the expressive nature of source code written by a group of computer scientists and hackers (including Richard Stallman), as well as the testimony by one of its authors, Carnegie Mellon Computer Science professor, David Touretzky, a fierce and well-known free speech loyalist. Just as they dissected Free Software licensing, F/OSS programmers quickly learned and dissected these courts cases, behaving in ways democratic theorists would no doubt consider exemplary. *Linux Weekly News,* for example, published the following overview and analysis of Touretzky's testimony:

> His [David Touretzky] point was that the restriction of source is equivalent to a restriction on speech, and would make it very hard for everybody who works with computers. The judge responded very well to Mr. Touretzky's testimony, saying things like . . . "*I think one thing probably has changed with respect to the constitutional analysis, and that is that subject to thinking about it some more, I really find what Professor Touretzky had to say today extremely persuasive and educational about computer code.*"
>
> [. . .] Thus, there are two rights being argued here. One is that . . . we have the right to look at things we own and figure out how they work. We even

> have the right to make other things that work in the same way. The other is that code is speech, that there is no way to distinguish between the two. In the U.S., of course, equating code and speech is important, because protections on speech are (still, so far) relatively strong. If code is speech, then we are in our rights to post it. If these rights are lost, Free Software is in deep trouble . . . [10]

In this exegesis we see again how Free Software developers wove together Free Software, source code, and free speech. These connections had recently been absent in hacker public discourse. Although Richard Stallman certainly grounded the politics of software in a liberal vocabulary of freedom, and Daniel Bernstein's fight introduced a far more legally sophisticated idea of the First Amendment for software, it was only with the DeCSS cases that a more prolific and specific language of free speech would come to dominate among F/OSS developers, and circulate beyond F/OSS proper. In the context of F/OSS development in conjunction with the DeCSS cases, the conception of software as speech became a cultural reality.

Much of the coherence emerged through reasoned political debate. Cleverness—or pranksmanship—played a pivotal part as well. Evan Prodromou, a Debian Developer and editor of one of the first Internet zines, *Pigdog,* circulated a decoy program that hijacked the name DeCSS, although it performed an entirely different operation from Johansen's DeCSS. Prodromou's DeCSS stripped Cascading Style Sheets (CSS) data (i.e., formatting information) from HTML pages:

> Hey, so, I've been really mad about the recent spate of horrible witch hunts by the MPAA against people who use, distribute, or even LINK TO sites that distribute DeCSS, a piece of software used for playing DVDs on Linux. The MPAA has got a bee in their bonnet about this DeCSS. They think it's good for COPYING DVDs, which, in fact, it's totally useless for. But they're suing everybody ANYWAYS, the bastardos!
> Anyways, I feel like I need to do something. I've been talking about the whole travesty here on Pigdog Journal and helped with the big flier campaign here in SF, . . . but I feel like I should do something more, like help redistribute the DeCSS software.
> There are a lot of problems with this, obviously. First and foremost, Pigdog Journal is a collaborative effort, and I don't want to bring down the legal shitstorm on the rest of the Pigdoggers just because I'm a Free Software fanatic.

> **DeCSS is Born**
> So, I decided that if I couldn't distribute DeCSS, I would distribute DeCSS. Like, I could distribute another piece of software called DeCSS, that is perfectly legal in every way, and would be difficult for even the DVD-CCA's lawyers to find fault with. [. . .]
> **Distribute DeCSS!**
> I encourage you to distribute DeCSS on your Web site, if you have one . . . I think of this as kind of an "I am Spartacus" type thing. If lots of people distribute DeCSS on their Web sites, on Usenet newsgroups, by email, or whatever, it'll provide a convenient layer of fog over the OTHER DeCSS. I figure if we waste just FIVE MINUTES of some DVD-CCA Web flunkey's time looking for DeCSS, we've done some small service for The Cause.[11]

Thousands of developers posted *Pigdog*'s DeCSS on their sites as flak to further confuse law enforcement officials and entertainment industry executives, who they felt were clueless about the nature of software technology. Dozens of them (including Jon Johansen) received cease-and-desist letters demanding they take down a version of DeCSS that was unrelated to the decryption DeCSS.

Clever re-creations of the original DeCSS source code (originally written in the C programming language) using other languages (such as Perl) also began to proliferate, as did translations into poetry, music, and film. A Web site hosted by David Touretzky, "The Gallery of CSS DeScramblers,"[12] showcased a set of 24 of these artifacts, the point being to demonstrate the difficulty of drawing a sharp line between functionality and expression in software. Touretzkey, an expert witness in the DeCSS case, said as much in the introductory statement to his Gallery:

> If code that can be directly compiled and executed may be suppressed under the DMCA, as Judge Kaplan asserts in his preliminary ruling, but a textual description of the same algorithm may not be suppressed, then where exactly should the line be drawn? This web site was created to explore this issue.

Here is a short snippet (about one-fifth), of the original DeCSS source code written in the C programming language:

```
void CSSdescramble(unsigned char *sec,unsigned char *key)
{
unsigned int t1,t2,t3,t4,t5,t6;
unsigned char *end=sec+0x800;
t1=key[0]^sec[0x54]|0x100;
```

```
t2=key[1]^sec[0x55];
t3=(*((unsigned int *)(key+2)))^(*((unsignedint *)(sec+0x56)));
t4=t3&7;
t3=t3*2+8-t4;
sec+=0x80;
t5=0;
while(sec!=end)
{
t4=CSStab2[t2]^CSStab3[t1];
t2=t1≫1;
t1=((t1&1)≪8)^t4;
t4=CSStab5[t4];
t6=(((((((t3≫3)^t3)≫1)^t3)≫8)^t3)≫5)&0xff;
t3=(t3≪8)|t6;
t6=CSStab4[t6];
t5+=t6+t4;
*sec++=CSStab1[*sec]^(t5&0xff);
t5≫=8;
}
```

Compare this fragment to another one written in Perl, a computer language that hackers regard as particularly well suited for crafting “poetic” code because longer expressions can be condensed into much terser, sometimes quite elegant (although sometimes quite obfuscated) statements. And indeed the original DeCSS program, composed of 9,830 characters, required only 530 characters in Perl:

```
#!/usr/bin/perl -w
# 531-byte qrpff-fast, Keith Winstein and Marc Horowitz
# <sipb-iap-dvd@mit.edu>
# MPEG 2 PS VOB file on stdin -> descrambled output on stdout
# arguments: title key bytes in least to most-significant order
$_=‘while(read+STDIN,$_,2048){$a=29;$b=73;$c=142;$t=255;@t=
map{$_%16or$t^=$c^=($m=(11,10,116,100,11,122,20,100)[$_/16%8])
&110;$t^=(72,@z=(64,72,$a^=12*($_%162?0:$m&17)),$b^=$_%64?
12:0,@z)[$_%8]}(16..271);if((@a=unx”C*”,$_)[20]&48){$h=5;$_=
unxb24,join”“,@b=map{xB8,unxb8,chr($_^$a[- - -$h+84])}@ARGV;s/
. . .   $/1$&/;$d=unxV,xb25,$_;$e=256|(ord$b[4])≪9|ord$b[3];$d=$d
≫8^($f=$t&($d≫12^$d≫4^$d^$d/8))≪17,$e=$e≫8^($t&($g=
($q=$e>>14&7^$e)^$q*8^$q<<6))<<9,$_=$t[$_]^(($h>>=8)+=$f
+(~$g&$t))for@a[128..$#a]}print+x"C*",@a}';s/x/pack+/g;eval
```

If Perl allows programmers to write code more poetically than other computer languages, Seth Schoen took up the challenge of publishing a bona fide poem in the form of an epic haiku—456 individual stanzas written over the course of just a few days. Schoen, who was inspired by the clever re-creations of DeCSS compiled in the gallery, wrote the poem to deliver a stark and clear political message. The author asserts that source code is not a metaphor or "similar to expression" but is expression, and he makes this point by recreating the original DeCSS program as a poem. This bit of poetry is now well known among hackers as an exemplary hack for it displays the cleverness that hackers collectively value.

The author opened his poem first by thanking Professor Touretzky and then moved immediately to abandon his "exclusive rights" clause of the copyright statute, indexing the direct influence of F/OSS licensing:

How to Decrypt a DVD: in haiku form
(Thanks, Prof. D. S. T.)
- - - - - - - - - - - - - - - - - - -
(I abandon my
exclusive rights to make or
perform copies of

this work, U. S. Code
Title Seventeen, section
One Hundred and Six.)

Muse! When we learned to
count, little did we know all
the things we could do

some day by shuffling
those numbers: Pythagoras
said "All is number"

long before he saw
computers and their effects,
or what they could do

by computation,
naive and mechanical
fast arithmetic.

It changed the world, it
changed our consciousness and lives
to have such fast math

available to
us and anyone who cared
to learn programming.

Now help me, Muse, for
I wish to tell a piece of
controversial math,

for which the lawyers
of DVD CCA
don't forbear to sue:

that they alone should
know or have the right to teach
these skills and these rules.

(Do they understand
the content, or is it just
the effects they see?)

And all mathematics
is full of stories (just read
Eric Temple Bell);

and CSS is
no exception to this rule.
Sing, Muse, decryption

once secret, as all
knowledge, once unknown: how to
decrypt DVDs.

Here, the author first frames the value of programming in terms of mathematics and its antagonists in the entertainment industry, IP statutes, lawyers, and judges, all of whom use software without recognizing, much less truly understanding, the embedded creative labor and expressive value. This critique is made explicit through a question: "Do they understand the content, or is it just the effects they see?"

The author then launches into a very long mathematical description of the forbidden CSS code represented in DeCSS. The expert explains the "player key" of CSS, which is the proprietary piece that enacts the access control measures:

> So this number is
> once again, the player key:
> (trade secret haiku?)
>
> Eighty-one; and then
> one hundred three—two times; then
> two hundred (less three)
>
> Two hundred and twenty
> four; and last (of course not least)
> the humble zero

The writer states the access control mathematically, but using words. From these lines alone a proficient enough programmer can deduce the encryption key. Thus the poem makes a similar point to the one made in the amicus brief, namely, that "[a]t root, computer code is nothing more than text, which, like any other text, is a form of speech. The Court may not know the meaning of the Visual BASIC or Perl texts . . . but the Court can recognize that the code is text."[13]

The author then conveys that many F/OSS programmers conceive of their craft as technically precise (and thus functional) yet fundamentally expressive, and as a result worthy of free speech protection. In formally comparing code to poetry in the medium of a poem, he displays a playful form of clever and recursive rhetoric valued among hackers (Fischer 1999); he also articulates both the meaning of the First Amendment and software to a general public:

> We write precisely
> since such is our habit in
> talking to machines;
>
> we say exactly
> how to do a thing or how
> every detail works.
>
> The poet has choice
> of words and order, symbols,
> imagery, and use

of metaphor. She
can allude, suggest, permit
ambiguities.

She need not say just
what she means, for readers can
always interpret.

Poets too, despite
their famous "license" sometimes
are constrained by rules:

How often have we
heard that some strange twist of plot
or phrase was simply

"Metri causa," for
the meter's sake, solely done
"to fit the meter"?

Although this haiku contains novel assertions (the tight coupling between source code and speech), it is also through its inscription into a tangible and especially a culturally captivating medium (a hack with playful and recursive qualities) that the assertion is transformed into firm social fact. Or, to put it another way, herewith a recondite legal argument makes its way into wide and public circulation and consumption. This is how discourse meant for public circulation, as theorist of publics Warner has noted, "helps to make a world insofar as the object of address is brought into being partly by postulating and characterizing it" (2002:91).

The protests, the poetry, and the debates demonstrate how programmers and hackers quickly became active participants in the drama of law and Free Software in the digital age. Together, they enact what legal theorist Robert Cover describes as a simultaneous process of subjective commitment to and objective projection of norms, a bridging that emerges out of a narrative mode. "This objectification of the norms to which one is committed frequently," Cover writes, "perhaps always entails a narrative—a story of how the law, now object, came to be, and more importantly, how it came to be one's own" (1992:145).

FREE DMITRY!

This narrative process, by which the law takes on a meaning to individuals through a period of contentious politics, would accelerate thanks to the simultaneous (although completely unrelated) DMCA infraction and arrest of another programmer, Dmitry Sklyarov. Because Sklyarov faced up to 25 years in jail, programmers in fact only grew more infuriated with the state's willingness to police technological innovation and software distribution through the DMCA. After Sklyrov's arrest, protest against the DMCA and the hacker commitment to a discourse of free speech only grew in emotional intensity and worked to extend the narrative process already underway.

This case would also prove far more dramatic than Johansen's because of the timing and place of the arrest. Sklyarov was arrested while leaving Defcon, the largest hacker conference. During the conference, Sklyarov had presented a paper on security breaches and weaknesses within the Adobe e-book format. He purportedly violated the DMCA by writing a piece of software for his Russian employer, Elcomsoft, that unlocks Adobe's e-book access controls and subsequently converts the files into the PDF format. For the FBI to arrest a programmer at the end of this conference was a potent statement. It showed that federal authorities would act on corporate demands to prosecute hackers under the DMCA. A description of the social and ritual significance of the hacker conference will make this clearer.

During Defcon, hackers spend four days denying their bodies' basic biological needs—notably sleep—so they can hack, party, and play with friends they usually interact with at length but largely only online. It is an intense, effervescent, and thoroughly ritual affair held yearly in America's strangest vacation playground, Las Vegas. Serious technical talks go hand-in-hand with parties, dancing, swimming, gambling, and games. Activities range from a three-day nonstop tournament in which teams attempt to crack an encrypted server, to suggestions like, "hey lets go check out Area 51 again." (Area 51 is the secret military base notorious for its shadowy "alien experiments.")

FBI agents attend this conference, but there is a well-known, although tacit, agreement that these agents, who are immediately identifiable by their L. L. Bean® khaki attire (normal Defcon regalia leans toward black clothing, T-shirts, and body piercings), not interfere with the hackers. Despite their presence since the 'con began in 1993, FBI agents had never arrested a hacker at this conference. (Typically, any arrests were local and because of excessively rowdy and drunken behavior.) The first ever FBI arrest of a hacker at Defcon sent a strong signal that intellectual property infractions were now serious criminal and federal offenses. This was a one-sided renegotiation of the relationship between legal authority and the hacker world.

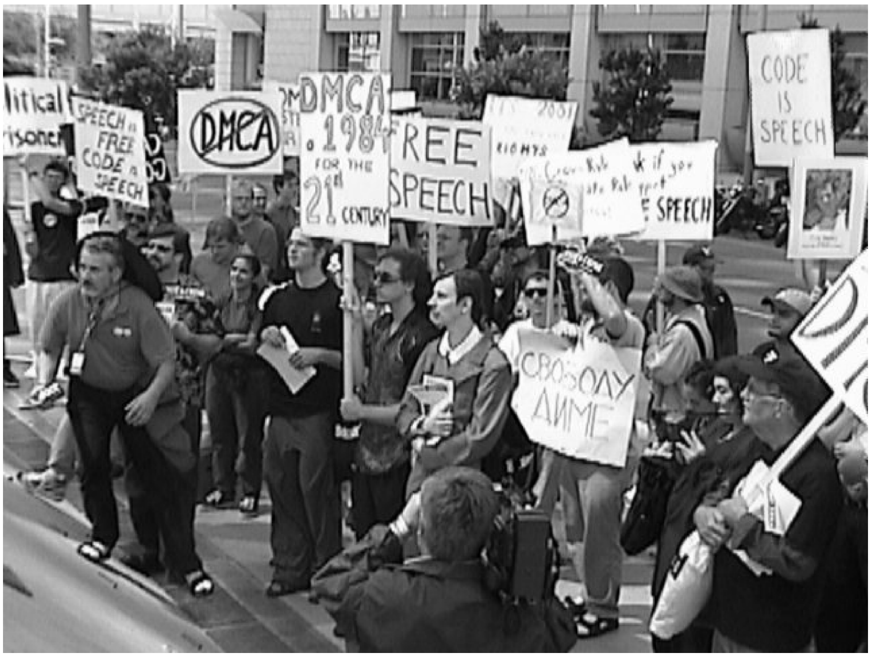

FIGURE 1. Free Dmitry Protest in San Francisco, California. Photograph by Ed Hintz.

On July 17, 2001, as Sklyarov was leaving the conference, federal agents whisked him away to an undisclosed jail in Nevada. Weeks later, he was released in the middle of a fervent "Free Dmitry" campaign. Sklyarov's arrest and related court hearings prompted discussions built on those initiated by Johansen's arrest and the resultant DeCSS lawsuits, but the Free Dmitry campaign was organized more swiftly, was more visible, and directly attacked Adobe, the company that had urged the Department of Justice to make the arrest. Developers organized protests across American cities (Boston, New York, Chicago, San Francisco, among others), in Europe, as well as Russia.

San Francisco, where at the time I was doing my fieldwork, was a hub of political mobilization. Even though Sklyarov was in no way identified with the world of F/OSS development, local F/OSS developers were behind a slew of protest activities, including a protest at Adobe's San Jose headquarters, a candlelight vigil at the San Jose public library, and a march held after Linux World on August 29, 2001, that ended up at the federal prosecutor's office (see Figure 1).

At a fund-raiser that followed the march to the prosecutor's office, Richard Stallman, the founder of the Free Software Foundation, and Lessig, the superstar activist-lawyer, gave impassioned speeches. Sklyarov, in a brief appearance, thanked the audience for their support. The mood was electric in an otherwise cool San Francisco warehouse loft. Lessig, who had recently published his *Code and Other Laws of Cyberspace,* a book that was changing the way F/OSS developers understood the politics of technology, fired up the already-animated crowd with charged declarations during his speech:

> Now this is America, right? It makes me sick to think this is where we are. It makes me sick. Let them fight their battles in Congress. These million-dollar lobbyists, let them persuade Congressmen about the sanctity of intellectual property and all that bullshit. Let them have their battles, but why lock this guy up for twenty-five years?

Most programmers agreed with Lessig's assessment: The state had gone way too far in its uncritical support of the copyright industries.

The protests had an immediate effect. Adobe withdrew its support of the case and, eventually, the court dropped all charges against Sklyarov on the condition that he testify in the subsequent case against his employers, which he did. In December 2002, the jury in that case acquitted Elcomsoft. Johansen was acquitted just over a year later because the charges against him were seen as far too shaky for prosecution (the law he was arrested under had nothing to do with digital rights management). Johansen still writes Free Software (including programs that subvert DRM technologies), as well as a blog, "So Sue Me," and is a hero among F/OSS hackers.

The DeCSS lawsuits were decided between 2001 and 2004, and even though the courts were persuaded that the DeCSS was a form of speech, they continued to uphold copyright law and deemed DeCSS unfit for First Amendment protection. In one of the *2600* cases, *Universal City Studios Inc. v. Reimerdes,* Judge Lewis A. Kaplan went so far as to declare that the court's decision meant to "contribute to a climate of appropriate respect for intellectual property rights in an age in which the excitement of ready access to untold quantities of information has blurred in some minds the fact that taking what is not yours and not freely offered to you is stealing."[14]

Developers and hackers were, in general, deeply disappointed with these decisions, which equated DeCSS with theft and were shocked about how narrow the consequences of Bernstein turned out to be. Many developers, however, emboldened and galvanized by the collective outpouring they organized or witnessed, continued to assert, in passionate and often considerable legal detail, a different narrative to that of piracy and stealing.[15] Indeed, these arrests, lawsuits, and protests helped establish as a cultural commonplace among F/OSS developers and hackers the connection between source code and speech. Hackers, programmers, and computer scientists continue to be motivated to transform what is now their cultural reality—a rival liberal morality—into a broader legal one by arguing that source code should be protectable speech under the U.S. Constitution and the constitutions of other nations.

CONCLUSION

Software developers have helped reconfigure central tenets of the liberal tradition—and specifically the meaning of free speech—to defend their productive autonomy. Many hackers, understood to be technologists, became legal thinkers and tinkerers, undergoing legal training in the context of the F/OSS project and building a corpus of liberal legal theory that links software to speech and freedom. By means of lively protest and prolific discussions, the connection between source code and speech was debated continuously between 1999 and 2003 by hackers, as well as new publics. It became a staple of Free Software moral philosophy and has helped add clarity in the competition between two different legal regimes (speech vs. IP) for the protection of knowledge and digital artifacts. Now other actors, such as activist lawyers, are consolidating new projects and body of legal work that challenge the shape and direction of IP law (Benkler 2006; Lessig 2001).

To be sure, the idea of free speech has never held a single meaning across the societies that have valued, instantiated, or debated it, but it has come to be seen as indispensable for a healthy democracy, a free press, individual self-development, and academic integrity. It is, as one media theorist has aptly put it, "as much cultural commonplace as an explicit doctrine" (Peters 2005:18; Streeter 2003). This pervasiveness makes the cultural analysis of liberal precepts, such as free speech, daunting (and always subject to important limitations), but is the reason why it deserves our attention. F/OSS, for example, is an ideal vehicle for examining how and when technological objects, such as source code, are reinvested with new liberal meanings, and with what consequences. By showing how developers incorporate legal ideals like free speech into the practices of everyday technical production, I trace the path by which older liberal ideals persist, albeit transformed, into the present.

This is key to emphasize, for even if we can postulate a relation between a product of creative work—source code—and a democratic ideal—free speech, there is no necessary or fundamental connection between them (Ratto 2005). Many academics and programmers have argued convincingly that the act of programming should be thought of as literary—"a culture of innovative and revisionary close reading" (Black 2002; see also Chopra and Dexter 2007). As with print culture of the last 200 years (Johns 2000), this literary culture of programming has often been dictated and delineated by a copyright regime whose logic is one of restriction. New free speech sensibilities, which fundamentally challenge the coupling between copyright and literary creation, must therefore be seen as a political act and choice, requiring sustained labor and creativity to stabilize these connections.

Hackers have been in part successful in this political fight because of their facility with the law; because of years of intensive technical training they have not only easily adopted the law but also tinkered with it to suit their needs. This active, transformative (and, one might say, populist) engagement with the law raises a set of pressing questions about the current state of global politics and legal advocacy. As Comaroff and Comaroff recently noted, the modern nation-state is one "rooted in a culture of legality" (2004:26), a culture that in recent years has become ever more pervasive, especially in the transnational arena. Whether it is the constitutional recognition of multiculturalism across Latin America and parts of Africa, or new avenues of commoditization like the patenting of seeds, these new political and economic relationships are "heavily inscribed in the language of the law" (Comaroff and Comaroff 2004:26). Given the extent to which esoteric legal codes dominate so many fields of endeavor, from pharmaceutical production to financial regulation to environmental advocacy, we must ask to what extent informal legal expertise, of the sort exhibited by F/OSS developers, is a necessary or useful skill for social actors seeking to challenge such regimes, and where and how advocates acquire legal literacy. We must remain alert to these amateur forms of legalism and to the alternative social forms that they imply. What this article suggests—indeed, what tracing out the relationship between hackers and the law suggests—is the extent to which the thing at issue in struggles over code is not only hackers' productive freedom but also the very meaning of democratic citizenship.

ABSTRACT

In this essay, I examine the channels through which Free and Open Source Software (F/OSS) developers reconfigure central tenets of the liberal tradition—and the meanings of both freedom and speech—to defend against efforts to constrain their productive autonomy. I demonstrate how F/OSS developers contest and specify the meaning of liberal freedom—especially free speech—through the development of legal tools and discourses within the context of the F/OSS project. I highlight how developers concurrently tinker with technology and the law using similar skills, which transform and consolidate ethical precepts among developers. I contrast this legal pedagogy with *more extraordinary legal battles over intellectual property, speech, and software. I concentrate on the arrests of two programmers, Jon Johansen and Dmitry Sklyarov, and on the protests they provoked, which unfolded between 1999 and 2003. These events are analytically significant because they dramatized and thus made visible tacit social processes. They publicized the challenge that F/OSS represents to the dominant regime of intellectual property (and clarified the democratic stakes involved) and also stabilized a rival liberal legal regime intimately connecting source code to speech.*

Keywords: law, open source, intellectual property, free speech, hacking

NOTES

Acknowledgments. The research for this essay was carried out between 2001 and 2004 and was supported by grants from the Social Science Research Council, the National Science Foundation, and the Charlotte W. Newcombe Foundation. Over the years, Chris Kelty, Genevieve Lakier, and Micah Anderson continually provided valuable feedback on this essay and I am very grateful for their support. I would also like to thank Seth Schoen and many other F/OSS developers (too many to list here) who provided me with feedback, support, and material for this work.

Versions of this essay were presented while I was a fellow at the Center for Cultural Analysis at Rutgers University, during the Law and Science workshop at Amherst College, and at the Franz Boas lecture at Columbia University. I would like to thank the participants at these events for their feedback, including Meredith McGill, Lisa Gitelman, Ellen Goodman, Greg Lastowka, Michael Warner, Paula McDowell, Marc Perlman, Martha Umphrey, Elizabeth Povinelli, Nadia Abu El-Haj, Samir Chopra, and Scott Dexter. Many others provided helpful comments on this work, pushing me to refine my arguments: Jean Comaroff, John Kelty, Gary Downey, Jelena Karanovic, Fred Benenson, Benjamin “Mako” Hill, Karl Fogel, Greg Pomerantz, Ben Kafka, Ben Peters, Alex Golub, Praveen Sinha, Alex Choby, Joe Hankins, James Grimmelmann, and Paul Berk. I am indebted to the five anonymous reviewers of this article and Mike and Kim Fortun for the invaluable feedback.

1. With this focus, I contribute to recent anthropological work that ties general political issues to a long history of debate about liberalism as lived reality (Comarroff and Comaroff 2003; Ong 2006; Povinelli 2002, 2006).
2. This comparison can only be made to do so much work. The law, being written in a natural language, contains all sorts of nuance, assumptions, and linguistic flexibility not present in the much more formal and rigid language of software; and of course, although programmers can acquire legal knowledge, they do not necessarily make good lawyers, a profession that requires many other skills on top of a formal comprehension of the law.
3. This was also the period when the counterglobalization protests were attacking the World Trade Organization who were leading the move to “harmonize” intellectual property law. Many of theses protests also put the issue of IP on the political map in new, more visible ways.
4. See http://www.eff.org/Privacy/Crypto_export/Bernstein_case/19990507_eff_pressrel.html, accessed November 14, 2008.
5. See Sewell (2005), Das (2003), Sahlins (1981), and Starr (2005) for examples of how exceptional historical events can work to stabilize and make visible new cultural connections and realities.
6. Despite the fact that one could us DeCSS to unlock copy controls on DVDs, the software cannot be used to make copies of DVDs.
7. Because of this preemptive feature, a number of scholars and lawyers critique this provision as draconian, for it obliterates the already fragile fair-use doctrine underpinning copyright law since 1976 (Gillespie 2007; Lessig 2001; Litman 2001; Samuelson 1999; Vaidhyanathan 2001, 2004).

8. See http://web.archive.org/web/20031124051048/cyber.law.harvard.edu/archive/dvd-discuss/msg00000.html, accessed November 10, 2008.
9. See http://slashdot.org/comments.pl?sid=3644&cid=1340340, accessed August 15, 2008.
10. See http://lwn.net/2000/0727/bigpage.php3, accessed November 20, 2008.
11. See http://www.pigdog.org/decss/, accessed February 5, 2009.
12. See http://www-2.cs.cmu.edu/∼dst/DeCSS/Gallery/, accessed November 10, 2008.
13. See http://cryptome.org/mpaa-v-2600-bac.htm, accessed April 23, 2009.
14. *Universal City Studios Inc. v. Reimerdes,* 82 F. Supplement 2d 211 [2000]. This case was appealed by one of the defendants, Eric Corley. In the subsequent case, *Universal City Studios Inc. v. Corley* 273 F. Supplement 3d 429 [2001], the presiding judges also affirmed the importance of this view in so far as they highlighted and quoted a longer version of this statement.
15. See *The History of the DeCSS Haiku* (Schoen 2004) for one of the most well-known examples.

Editors' Notes: Cultural Anthropology has published a number of essays on the practices and politics of informationalism. See particularly Brian Axel's "Anthropology and the New Technologies of Communication" (2006), Christopher Kelty's "Geeks, Social Imaginaries, and Recursive Publics" (2005), and René T. A. Lysloff's "Musical Community on the Internet: An On-Line Ethnography" (2003). Also see Anthropology of/in Circulation: The Future of Open Access and Scholarly Societies, a conversation in *Cultural Anthropology* amongst open access advocates, accessible online at: http://blog.culanth.org/incirculation/.

Cultural Anthropology has also published a number of essays on the politics of law. See Damani Partridge's "We Were Dancing in the Club, Not on the Berlin Wall: Black Bodies, Street Bureaucrats, and Exclusionary Incorporation into the New Europe" (2008), Heather Paxson's "Post-Pasteurian Cultures: The Microbiopolitics of Raw-Milk Cheese in the United States" (2008), Ilana Feldman's "Difficult Distinctions: Refugee Law, Humanitarian Practice, and Political Identification in Gaza" (2007), and Sarah Jain's "'Dangerous Instrumentality': The Bystander as Subject in Automobility" (2004).

REFERENCES CITED

Axel, Brian
2006 Anthropology and the New Technologies of Communication. Cultural Anthropology 21(3):354–384.

Benkler, Yochai
2006 The Wealth of Networks. New Haven, CT: Yale University Press.

Black, Maurice J.
2002 The Art of Code. Ph.D. dissertation, Department of English, University of Pennsylvania.

Castells, Manuel
2001 The Internet Galaxy. Oxford: Oxford University Press.

Chopra, Samir, and Scott Dexter
2007 Decoding Liberation: The Promise of Free and Open Source Software. New York: Routledge.

Coleman, E. Gabriella
2004 The Political Agnosticism of Free and Open Source Software and the Inadvertent Politics of Contrast. Anthropology Quarterly 77(3):507–519.

Coleman, E. Gabriella, and Alex Golub
2008 Hacker Practice: Moral Genres and the Cultural Articulation of Liberalism. Anthropological Theory 8(3):255–277.

Cohn, Cindy, and James Grimmelmann
2003 Seven Ways in which Code Equals Law. Electronic document, http://90.146.8.18/en/archives/festival_archive/festival_catalogs/festival_artikel.asp?iProjectID=12315, accessed January 27, 2009.

Comaroff, Jean, and John Comaroff
2003 Reflections on Liberalism, Policulturalism, and ID-Ology: Citizenship and Difference in South Africa. Social Identities 9(3):445–474.

2004 Criminal Obsessions, after Foucault: Postcoloniality, Policing, and the Metaphysics of Disorder. Critical Inquiry (30):800–824.

Cover, Robert

1992 Nomos and Narrative. *In* Narrative, Violence, and the Law: The Essays of Robert Cover. Martha Minow, Michael Ryan, and Austin Sarat, eds. Pp. 95–172. Ann Arbor: University of Michigan Press.

Das, Veena

2003 Critical Events. Oxford: Oxford University Press.

Ewick, Patricia, and Susan Silbey

1998 The Common Place of the Law. Chicago: University of Chicago Press.

Feldman, Ilana

2007 Difficult Distinctions: Refugee Law, Humanitarian Practice, and Political Identification in Gaza. Cultural Anthropology 22(1):129–169.

Fischer, Michael

1999 Worlding Cyberspace: Towards a Crucial Ethnography in Time, Space, Theory. *In* Critical Anthropology Now: Unexpected Context, Shifting Constituencies, Changing Agendas. George Marcus ed. Pp. 245–304. Santa Fe: SAR Press.

Fortun, Kim

2001 Advocacy after Bhopal: Environmentalism, Disaster, New Global Orders. Chicago: University of Chicago Press.

Galloway, Alexander R.

2004 Protocol: How Control Exists after Decentralization. Cambridge, MA: MIT Press.

Gillespie, Tarleton

2007 Wired Shut: Copyright and the Shape of Digital Culture. Cambridge, MA: MIT Press.

Gupta, Akhil, and James Ferguson

1997 Discipline and Practice: “The Field” as Site, Method, and Location in Anthropology. *In* Anthropological Locations: Boundaries and Grounds of a Field Science. Akhil Gupta and James Ferguson, eds. Pp. 1–46. Berkeley: University of California Press.

Hesse, Carla

2002 The Rise of Intellectual Property, 700 B.C.–A.D. 2000: An Idea in the Balance. Daedalus (Spring):6–45.

Himanen, Pekka

2001 The Hacker Ethic and the Spirit of the Information Age. New York: Random House.

Jain, Sarah

2004 “Dangerous Instrumentality”: The Bystander as Subject in Automobility. Cultural Anthropology 19(1):61–94.

Johns, Adrian

2006 Intellectual Property and the Nature of Science. Cultural Studies 20(2–3):145–164.

2000 The Nature of the Book. Chicago: University of Chicago Press.

Kelty, Chris

2005 Geeks, Social Imaginaries, and Recursive Publics. Cultural Anthropology 20(2):185–214.

2008 Two Bits: The Cultural Significance of Free Software and the Internet. Durham, NC: Duke University Press.

Knuth, Donald

1998 The Art of Computer Programming. 3 vols. New York: Addison-Wesley.

Latour, Bruno

1988 Pasteurization of France. Alan Sheridan and John Law, trans. Cambridge, MA: Harvard University Press.

Litman, Jessica
2001 Digital Copyright: Protecting Intellectual Property on the Internet. Amherst, NY: Prometheus.

Lessig, Lawrence
1999 Code and Other Laws of Cyberspace. New York: Perseus.
2001 The Future of Ideas: The Fate of the Commons in a Connected World. New York: Random House.

Lysloff, René T. A.
2003 Musical Community on the Internet: An On-Line Ethnography. Cultural Anthropology 18(2):233–263.

Macpherson, C. B.
1962 The Political Theory of Possessive Individualism: Hobbes to Locke. Oxford: Clarendon Press.

Mezey, Naomi
2001 Law as Culture. Yale Journal of Law and the Humanities 13(1):35–68.

McGill, Meredith
2002 American Literature and the Culture of Reprinting, 1834–1853. Philadelphia: University of Pennsylvania Press.

O'Mahony, Siobhán, and Fabrizio Ferraro
2007 The Emergence of Governance in an Open Source Community. Academy of Management Journal 50(5):1079–1106.

Ong, Aihwa
2006 Neoliberalism as Exception: Mutations in Citizenship and Sovereignty. Durham, NC: Duke University Press.

Partridge, Damani
2008 We Were Dancing in the Club, Not on the Berlin Wall: Black Bodies, Street Bureaucrats, and Exclusionary Incorporation into the New Europe. Cultural Anthropology 23(4):660–687.

Paxson, Heather
2008 Post-Pasteurian Cultures: The Microbiopolitics of Raw-Milk Cheese in the United States. Cultural Anthropology 23(1):15–47.

Peters, John Durhman
2005 Courting the Abyss: Free Speech and the Liberal Tradition. Chicago: University of Chicago Press.

Pfaffenberger, Bryan
1996 "If I Want It, It's OK": Usenet and the (Outer) Limits of Free Speech. Information Society 12:365–388.

Povinelli, Elizabeth
2002 The Cunning of Recognition: Indigenous Alterities and the Making of Australian Multiculturalism. Durham, NC: Duke University Press.
2006 The Empire of Love: Toward a Theory of Intimacy, Genealogy, and Carnality. Durham, NC: Duke University Press.

Ratto, Matt
2005 Embedded Technical Expression: Code and the Leveraging of Functionality. Information Society 21(3):205–213.

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

Riemens, Patrice
2003 Some Thoughts on the Idea of "Hacker Culture." Multitudes 8(2). Electronic document, http://multitudes.samizdat.net/Some-thoughts-on-the-idea-of.html, accessed January 27, 2009.

Sahlins, Marshall
1981 Historical Metaphors and Mythical Realities: Structure in the Early History of the Sandwich Islands Kingdom. Ann Arbor: University of Michigan Press.

Salin, Peter
1991 Freedom of Speech in Software. Electronic document, http://www.philsalin.com/patents.html, accessed November 12, 2003.
Samuelson, Pamela
1999 Intellectual Property and the Digital Economy: Why the Anti-Circumvention Regulations Need to Be Revised. Berkeley Tech Law Journal 14:519–548.
Sewell, William
2005 Logics of History: Social Theory and Social Transformation. Chicago: University of Chicago Press.
Schoen, Seth
2001 How to Decrypt a DVD: In Haiku Form. Electronic document, http://www.cs.cmu.edu/∼dst/DeCSS/Gallery/decss-haiku.txt, accessed October 2, 2008.
2004 The History of the DeCSS Haiku. Electronic document, http://www.loyalty.org/∼schoen/haiku.html, accessed January 10, 2008.
Starr, Paul
2005 The Creation of the Media: Political Origins of Modern Communication. New York: Basic.
Streeter, Thomas
2003 The Romantic Self and the Politics of Internet Commercialization. Cultural Studies 17(5):648–668.
Thomas, Douglas
2002 Hacker Culture. Minneapolis: University of Minnesota Press.
Tilly, Charles, and Sidney Tarrow
2006 Contentious Politics. Boulder, CO: Paradigm.
Vaidyanathan, Siva
2001 Copyrights and Copywrongs: The Rise of Intellectual Property and How It Threatens Creativity. New York: New York University Press.
2004 The Anarchist in the Library: How the Clash between Freedom and Control Is Hacking the Real World and Crashing the System. New York: Basic.
2006 Critical Information Studies: A Bibliographic Manifesto. Cultural Studies 20(2–3):292–315.
Wark, McKenzie
2004 A Hacker Manifesto. Cambridge, MA: Harvard University Press.
Warner, Michael
2002 Publics and Counterpublics. New York: Zone.
Weber, Steven
2004 The Success of Open Source. Cambridge, MA: Harvard University Press.
Yngvesson, Barbara
1989 Inventing Law in Local Settings—Rethinking Popular Legal Culture. Yale Law Journal 98(8):1689–1709.