**GREG R. VETTER**[*]

# Exit and Voice in Free and Open Source Software Licensing: Moderating the Rein over Software Users

CONTENTS



---

[*] Assistant Professor of Law, University of Houston Law Center (UHLC); Codirector, Institute for Intellectual Property and Information Law (IPIL). Biography and additional background available at: http://www.law.uh.edu/ faculty/gvetter. Relevant to this work is that my background includes a Master's degree in computer science and full-time employment experience in the business-to-business software industry from 1987 to 1996. For helpful comments and discussion, I thank Gabriella Coleman, Bob Gomulkiewicz, Paul Janicke, Craig Joyce, Ray Nimmer, Joel West, Peter Yu, and the participants at the Works-in-Progress Intellectual Property Colloquium 2005 cosponsored by the Washington University School of Law and Saint Louis University School of Law. Research for this work was supported by a summer research grant from the University of Houston Law Foundation. I also thank UHLC's IPIL Institute and its sponsors for support of my endeavors at UHLC. Many thanks to UHLC students Stacey Reese and Nivine Zakhari for their research assistance. In addition, I am thankful for the exceptionally capable support provided by the staff of the University of Houston Law Center's John M. O'Quinn Law Library.

Passionate politics and contrarian economics pervade free and open source software (FOSS). Symbiotically, these hallmark characteristics of FOSS need each other. Reflecting concepts of voice and exit, they jointly emerged in a cooperation that gives FOSS part of its motive force. Voice, the expression of FOSS's functional freedom for computer users, corresponds to political effects. Exit, the choice by a proprietary software user to switch to FOSS, corresponds to economic effects. Voice has two forms: direct, where a user complains to the software provider, and indirect, where a user complains generally through means such as advocacy, evangelism, or lobbying. These politics and economics express their energetic and productive tension through FOSS code and licensing, and through FOSS stakeholders within the software ecosystem.

The interplay of exit and voice is a crucial, yet thus far underappreciated, element in the FOSS phenomenon. This Article analyzes four situations from the FOSS movement. It proposes that a full understanding of FOSS includes the perspectives arising from the interplay between exit and voice. The analysis applies the framework in Albert O. Hirschman's book *Exit, Voice, and Loyalty*.[1] Hirschman observed that mechanisms of both exit and voice discipline an organization that allows its products or services to degrade to a state of user dissatisfaction or lesser quality, as in, for example, a disgruntled employee, who might quit (exit), or complain (voice).[2] The exit mechanism corresponds with the economic approach as a disciplining force, while voice corresponds with the political or sociological approach.[3] Hirschman did not elevate one mechanism over the other. He noted the interplay between them for a variety of conditions and institutions, introducing loyalty as an interposing mechanism that allows voice room to operate.[4] His framework has been influential over the last thirty-five years in a variety of legal fields,[5] and more generally across the social sciences.[6]

---

1 ALBERT O. HIRSCHMAN, EXIT, VOICE, AND LOYALTY: RESPONSES TO DECLINE IN FIRMS, ORGANIZATIONS, AND STATES (1970).

2 *Id.* at 3-4.

3 *Id.* at 15-16.

4 *Id.* at 1-4, 15-18, 77-80.

5 *See, e.g.*, Samuel Bacharach & Peter Bamberger, *The Power of Labor to Grieve: The Impact of the Workplace, Labor Market, and Power-Dependence on Employee Grievance Filing*, 57 INDUS. & LAB. REL. REV. 518, 519 (2004) (using Hirschman's concepts of "exit," "voice," and "loyalty" to derive a model for employee grievance filing); Larry Cata Backer, *Surveillance and Control: Privatizing and Nationalizing Corporate Monitoring After Sarbanes-Oxley*, 2004 MICH. ST. L. REV. 327, 348 (discussing how the SEC's shareholder model allows for more shareholder "voice"); Margaret M. Blair, *Reforming Corporate Governance: What History Can Teach Us*, 1 BERKELEY BUS. L.J. 1, 4, 7-8, 32, 39 (2004) (discussing how corporate governance reform can affect a shareholder's "voice" and "exit"); Theodore Eisenberg & Geoffrey Miller, *The Role of Opt-Outs and Objectors in Class Action Litigation: Theoretical and Empirical Issues*, 57 VAND. L. REV. 1529, 1539 (2004) (citing John C. Coffee, Jr., *Class Action Accountability: Reconciling Exit, Voice, and Loyalty in Representative Litigation*, 100 COLUM. L. REV. 370, 377 (2000) and Samuel Issacharoff, *Governance and Legitimacy in the Law of Class Actions*, 1999 SUP. CT. REV. 337, 366); Ken Matheny & Marion Crain, *Disloyal Workers and the "Un-American" Labor Law*, 82 N.C. L. REV. 1705, 1705 (2004) (examining "exit," "voice," and "loyalty" and their role in work relations).

6 *See, e.g.*, Alison Davis-Blake et al., *Happy Together? How Using Nonstandard Workers Affects Exit, Voice, and Loyalty Among Standard Employees*, 46 ACAD. MGMT. J. 475, 475 (2003) (examining how a "blended workforce . . . affected exit, 'voice,' and

Exit and voice marry in unique ways in FOSS. Policy and legal choices including FOSS licensing issues, FOSS license proliferation, and estimations of where FOSS fits best in the software ecosystem should take account of the symbiotic interplay of exit and voice in FOSS.

The passionate politics come from the free software advocates within the movement. For this group, self-determination and functional freedom with one's computer is the goal. The free software advocates designed a counterintuitive copyright-based licensing system that demands preservation of the right to share software in a form that promotes functional freedom for computer users. The central preserving conditions are that: (1) the source code is available and (2) no one is charged royalties for ongoing use. Combine these with: (3) a right to redistribute in modified or unmodified form and (4) the requirement that redistributed software reapply these conditions, and you have a self-perpetuating licensing system that preserves software freedom for copies or generational derivatives.[7]

Politics spawned this counterintuitive license; specifically, it was the clash of Richard Stallman's politics with the corporate practice to make software source code secret. Stallman was part of an early com munity of programmers at the Massachusetts Institute of Technology (MIT), where developers freely shared code, and where he developed his philosophical approach to software sharing.[8] As computers became more important to business, however, corporate

---

loyalty among standard employees"); Donald W. Light et al., *No Exit and the Organization of Voice in Biotechnology and Pharmaceuticals*, 28 J. HEALTH POL. POL'Y & L. 473, 474 (2003) (discussing the concept of organized "voice" in the biotechnology and pharmaceuticals fields); Richard E. Matland, *Exit, Voice, Loyalty and Neglect in an Urban School System*, 76 SOC. SCI. Q. 506 (1995); Mary Jo Bane, *Exit, Voice and Loyalty in the Church*, AMERICA, June 3, 2002, at 12, 14 (discussing how members of the Catholic Church will react to the sex crime scandals).

[7] *See* GNU Project, GNU General Public License Version 2, http://www.gnu.org/licenses/gpl.txt (last visited Sept. 15, 2006) [hereinafter GPL]. Version three of the GPL was posted in draft form in January 2006 to initiate a public revision process for the license. *See* GPLv3 Draft, http://gplv3.fsf.org/gpl-draft-2006-01-16.html (last visited Sept. 15, 2006) [hereinafter GPLv3]. *See also* David McGowan, *Legal Implications of Open-Source Software*, 2001 U. ILL. L. REV. 241, 253-60 (reviewing common terms in open source and free software licenses); Greg R. Vetter, *The Collaborative Integrity of Open-Source Software*, 2004 UTAH L. REV. 563, 599 (describing the open source approach taken by the GPL).

[8] *See* RICHARD M. STALLMAN, *The GNU Project*, *in* FREE SOFTWARE, FREE SOCIETY: SELECTED ESSAYS OF RICHARD M. STALLMAN 15-16 (2002) (recounting Stallman's time working in the MIT Artificial Intelligence Lab during the 1970s).

approaches impinged on the community norms. Source code, from the corporate perspective, became a valuable asset. Under the corporate mindset, source code fits under trade secrets because a company could profit from the object code, either by renting or licensing it, without disclosing the source code.[9] Stallman decided to exit these corporate influences as abhorrent to his politics.[10] He invented free software licensing, encoding its terms in the General Public License (GPL). His goal was to "to guarantee [the] freedom to share and change free software."[11]

The licensing norms spawned by Stallman's politics also engendered contrarian economics for FOSS: use of the software was free. A camp of pragmatists following in the wake of the free software advocates emphasized that FOSS supports markets even though the software itself is royalty-free as to its use. These pragmatists became known as the open source software camp. They are identified with a different emphasis within FOSS: creating high-quality software and building a user base for it.[12]

FOSS licensing, with its emphasis on source code availability, facilitated a new collaborative software development model. The model has churned out a number of important software applications, including the computer operating system known as GNU/Linux.[13] It also generated the Apache web server, the market leader for

---

9 *See* RAYMOND T. NIMMER, THE LAW OF COMPUTER TECHNOLOGY: RIGHTS, LICENSES, LIABILITIES § 1:37 (2d ed. Supp. 2005).

10 *See* STALLMAN, *supra* note 8, at 17 (describing the "stark moral choice" Stallman was facing when he decided to start his FOSS project creating an open source operating system).

11 GPL, *supra* note 7, at Preamble.

12 *See* LINUS TORVALDS & DAVID DIAMOND, JUST FOR FUN: THE STORY OF AN ACCIDENTAL REVOLUTIONARY 163-71 (2001).

13 The GNU/Linux operating system is sometimes referred to as Linux. An operating system, however, is not a single large software work, but is rather an aggregation of many software components. The central component is the kernel, which is properly called Linux. Distributions of a Linux kernel-based operating system include other critical components. Most distributions include a set of essential software tools from the GNU project, a separate open source software effort. Richard Stallman, The GNU Project, http://www.gnu.org/gnu/ thegnuproject.html (found under the heading "Linux and GNU/Linux") (last visited June 8, 2006). Thus, some use the name "GNU/Linux" for such a distribution. *Id.* ("We call this system version GNU/Linux, to express its composition as a combination of the GNU system with Linux as the kernel.") The GNU acronym is a self-referential label meaning "GNU's Not UNIX," with Unix being a predecessor computer operating system. *See* The GNU Operating System, http://www.gnu.org (last visited Aug. 9, 2006).

delivering web pages to browsers over the Internet.[14] These FOSS products compete with proprietary products from commercial software providers. Indeed, the open source software camp is willing to embrace certain types of commercialization in a way that free software advocates might not. They are generally more tolerant of corporate influences. In fact, the open source camp used these influences to establish relationships through which it could spread the benefits of FOSS collaborative development and of sharing source code to gather powerful allies for the movement.[15]

The economics of FOSS licensing enables collaborative development and complementary corporate opportunity. The licensing system's prohibition on royalties was intended to preserve the ability to share the software and its source code.[16] Such a prohibition sets a price of zero to run the software, an attractive level when market conditions are ready to accept the software as valuable. Most prominent among the open source advocates, Linus Torvalds provided the impetus for greater acceptance of FOSS. He started an operating system kernel and then accepted collaborators from across the globe to enable FOSS's flagship application, the GNU/Linux operating system.[17] Torvalds describes himself as uninterested in politics.[18] His motivation is to generate quality software.[19] The

---

[14] *See* Netcraft, July 2005 Web Server Survey, http://news.netcraft.com/archives/2005/07/01/july_2005_web_server_survey.html (reporting a 69.8% market share for Apache products on active web sites in July and 22.8% for Microsoft, the next most popular provider).

[15] *See* Jim Hamerly & Tom Paquin, *Freeing the Source: The Story of Mozilla*, *in* OPEN SOURCES: VOICES FROM THE OPEN SOURCE REVOLUTION 197, 203-06 (Chris DiBona et al. eds., 1999) (describing the events leading up to Netscape's decision to release the source code for its web browser, Mozilla). *See also* Michael Paige, *IBM Gives Database Code to Open-Source Community*, INVESTOR'S BUS. DAILY, Aug. 3, 2004, http://www.investors.com/breakingnews.asp?journalid =22493986&brk=1 (preannouncing IBM's donation of database source code valued at an estimated $85 million to the open source community).

[16] Stallman, *supra* note 13, at "Copyleft and the GNU-GPL."

[17] *See* Linus Torvalds, *The Linux Edge*, *in* OPEN SOURCES, *supra* note 15, at 101-02, 108-11 (describing the rationale behind his decisions at Linux's inception and through the initial stages of its development).

[18] *See* TORVALDS & DIAMOND, *supra* note 12, at 165 (explaining his preference for the more conciliatory European political system as opposed to the more combative American style); Interview by Marjorie Richardson with Linus Torvalds (Nov. 1, 1999), http://interactive.linuxjournal.com/node/3655 (responding to queries about his political interests, Torvalds said "I'm absolutely uninterested in politics. . . . I really don't want to go into politics").

FOSS licensing approach establishes ground rules for an innovative, distributed software development approach based on collaboration.[20] The prohibition on royalties means that contributing developers do not have to worry about licensing costs. In addition, it attracts distributors.

A well-known distributor of GNU/Linux is Red Hat.[21] Another prominent company commercializing FOSS is IBM.[22] GNU/Linux and a host of other open source products complement IBM's hardware and services business. Both Red Hat and IBM facilitate exit from proprietary software to FOSS. The FOSS prohibition on royalties enables their corporate opportunity in FOSS services.

The free software advocates brought forth FOSS with a passion that rings with "voice," as that term is used in *Exit, Voice, and Loyalty*. In Hirschman's account of the forces that may recuperate a decline in quality or aptitude in a firm, state, or organization, voice corresponds to the political or social functioning of the firm, state, or organization.[23] Focusing on Hirschman's thesis as it applies to a firm, customers who call or write letters to complain about a product, but who do not switch to a different product, exercise voice.[24]

---

[19] *See* Torvalds, *supra* note 17, at 111 (stating that he "want[s] Linux to be on the cutting edge, and even a bit past the edge, because what's past the edge today is what's on your desktop tomorrow").

[20] Everyone can see the source code, so remote developers can contribute. Moreover, ubiquitous source code may procedurally enhance software quality: all developers and users can see the code, resulting in "massive peer review" to generally increase software quality and defeat bugs. ERIC S. RAYMOND, THE CATHEDRAL AND THE BAZAAR: MUSINGS ON LINUX AND OPEN SOURCE BY AN ACCIDENTAL REVOLUTIONARY 4 (1999), *available at* http://www.catb.org/~esr/ writings/cathedral-bazaar/cathedral-bazaar/ (coining the phrase "given enough eyeballs, all bugs are shallow," also known as "Linus's Law").

[21] *See* Red Hat, The Open Source Leader, http://www.redhat.com/about/ (last visited June 8, 2006) ("Today Red Hat is the world's most trusted provider of Linux and open source technology."). *See also* William M. Bulkeley, *Can Linux Take Over the Desktop? Open-Source Software Is Ready to Do Battle on a New Front; Here's a Look at Its Chances*, WALL ST. J., May 24, 2004, at R1 (characterizing Red Hat as "the leading U.S. distributor of Linux").

[22] IBM, Open Source, Resources for Open Source Development and Implementation, http://www-128.ibm.com/developerworks/opensource (last visited June 8, 2006) (providing a forum for the open source community and updates regarding open source development).

[23] HIRSCHMAN, *supra* note 1, at 4, 15-16.

[24] *Id.* at 4, 30, 36-37.

Switching products, on the other hand, is a choice for exit.[25] In *Exit, Voice, and Loyalty*, Hirschman discusses factors influencing which of these two disciplining forces may be more effective in a given situation.[26] Economists often view exit as the superior option, but exit is not always practically available; in cases where it is not, voice plays a more important role.[27] Often, both forces are at work in varying degrees because customers have differing sensibilities.[28]

This Article applies Hirschman's framework in *Exit, Voice, and Loyalty* to FOSS, focusing on the context of corporate users of proprietary software, in which FOSS alternatives provide a unique exit opportunity cloaked in direct and indirect voice.[29] I use the label "direct voice" to refer to Hirschman's paradigmatic voice example: customer complaints to the supplier, calling or hoping for a remedy.[30] The label "indirect voice" moves away from a specific target supplier as audience.[31] It includes group behavior, norm evangelism, advocacy (using public channels or legal forums), and

---

[25] *Id.* at 4, 36-37.

[26] *See id.* at 36-37, 43.

[27] *Id.* at 21, 33, 43, 80, 83.

[28] *See id.* at 17-18, 22-25, 36-37, 48-49, 77, 80, 83, 111, 124.

[29] Hirschman's framework has been applied to intellectual property and information law issues in a few related instances. *See, e.g.*, Dan L. Burk, *Virtual Exit in the Global Information Economy*, 73 CHI.-KENT L. REV. 943, 945 (1998) (arguing that the Internet affords an "unprecedented opportunity to explore the interplay of" exit and voice for digital goods, and that lower cost exit spurns market effects that "will facilitate competition among firms for information products, and so among nations for intellectual property regulation"); Light et al., *supra* note 6, at 475-77, 497 (utilizing "exit" and "voice" theories in the context of pharmaceutical companies who, the authors argue, created a situation of minimal exit from their products and noting that the companies may also corrupt the voice channels used by those entrapped); Dawn C. Nunziato, *Exit, Voice, and Values on the Net*, 15 BERKELEY TECH. L.J. 753, 754, 758-60 (2000) (discussing the "preference-expressing mechanisms of exit and voice" and their interplay with regulation of the Internet in reviewing LAWRENCE LESSIG, CODE AND OTHER LAWS OF CYBERSPACE (1999)); Tim Wu, *When Code Isn't Law*, 89 VA. L. REV. 679, 696 & n.58, 697-709 (2003) (comparing exit and voice to dichotomous mechanisms to change, or avoid, laws regulating information goods, where by applying the interest group work of Mancur Olson, peer-to-peer file sharing is a collective-action mechanism to allow certain groups to avoid—i.e., exit—copyright law).

[30] HIRSCHMAN, *supra* note 1, at 4, 16, 30.

[31] *See* Michael Laver, "*Exit, Voice, and Loyalty" Revisited: The Strategic Production and Consumption of Public and Private Goods*, 6 BRIT. J. POL. SCI. 463, 464-69, 473-74 (1976) (discussing Hirschman's model and arguments by other scholars who deemphasize the voice mechanism). Furthermore, Laver argues from a rational choice perspective that the value of voice is tied to the threat of exit, and, more importantly, to voice as feedback, i.e., the possibility of informing others about the decline in quality both before and after exit, thus engendering exit beyond the sole speaker. *Id.*

lobbying.[32] While these two types of voice are related, and both may spring from the same message, these two categories structure my analysis.[33]

The origination of free software by Stallman was cloaked in activism. He perceived that software is inherently of insufficient quality when the source code is not available or shareable, because in these situations one cannot revise the code or have others revise it.[34] He expressed his view and his indirect voice attracted many followers. In effect, his "exit"[35] to FOSS was "noisy" with indirect voice because it was tinged, at least in part, by his political perspective that full self-determination with one's computer is a fundamental freedom. In this view, even if one cannot reprogram the software herself, the opportunity to do so, or to pay someone else to reprogram it, is critical. The establishments providing this "low-quality" software are any individual or entity distributing software without source code and with licensing terms that prohibit free sharing.[36]

---

[32] One purpose of this Article is to initially explore the possibility of voice-favoring by decision makers within the framework of *Exit, Voice, and Loyalty* in the context of FOSS. This Article puts aside First Amendment theory. However, I acknowledge that First Amendment theory offers other, potentially more important reasons for voice-favoring. The reasons flowing from the exit and voice framework are additive.

[33] The literature shows classifications of voice similar to my direct-versus-indirect bifurcation. *See, e.g.*, Light et al., *supra* note 6, at 477-78 & fig.1 (noting that scholars have tended to distinguish between the two types of voice, referring to them as "vertical" and "horizontal"). Specifically, the authors describe Hirschman's voice mechanism as "vertical voice" (i.e., the suppliers) and "horizontal voice" as organization of the dissatisfied vertical speakers (i.e., the customers). *Id.* Further, horizontal voice is said to suffer from various coordination and collective action problems. *Id.*

[34] *See* STALLMAN, *supra* note 8, at 119-32 (emphasizing the overall social benefits of unrestricted access to source code, including greater user ability to evolve applications).

[35] I put "exit" in quotes because Stallman was never a proprietary software user. Stallman, *supra* note 13, at "The First Software-Sharing Community." In that sense, he never exited. In another sense, he exited the software world he inhabited when he was faced with the prospect of that world no longer sharing source code. *Id.* at "The Collapse of the Community." In Hirschman's framework, the quality of his experience was about to change, and exit was the preferable option after voice had failed. HIRSCHMAN, *supra* note 1, at 36-37.

[36] Software distribution without source code and without the right to share it describes virtually the entire proprietary software products industry, except for some software component products. Common examples include software companies like Microsoft, Oracle, Corel, or Computer Associates, and companies with significant hardware and software revenues like IBM, Sun, HP, and Apple. It also describes most in-house software development when confronted with an opportunity to distribute software to third parties.

To examine these themes, Part I discusses the FOSS exit alternative for various software user categories in order to illustrate the dynamics a customer faces when considering the switch from proprietary software to FOSS.[37] For a few FOSS products, the signs that user exit is having some disciplining effect, as contemplated in *Exit, Voice, and Loyalty*, are unmistakable. For example, in response to FOSS, Microsoft implemented its "Shared Source Initiative" program where it allows developers to review the source code for some of its software.[38]

Part II explores in more detail the licensing terms defining FOSS exit from proprietary software and relates these to the voice expressed in the license. While all FOSS licenses define an exit opportunity, the licenses vary in the degree to which they express indirect voice. Voice content is sometimes found in a FOSS license, but is more often found in related materials, such as the web site where the license is located.[39] With its institutional mixture of exit and indirect voice, the FOSS license enables users to exit from proprietary software for some applications, and that very act becomes direct voice by the user toward the vendor for other applications. This partial exit has a direct voice effect and makes more credible future threats of exit in other applications.

This Part also describes the FOSS user's situation after switching to FOSS. This is important because the estimation of that situation helps the user decide whether to make the leap. FOSS has different development team transparency and user participation opportunities compared to traditional software. These inform the character of the FOSS exit because an ongoing relationship often underlies the connection between many corporate users and their software suppliers. In traditional software, the ongoing relationship is typically based on a contract, but often supplemented by noncontractual communication. In FOSS, the relationship occurs

---

37 STEVEN WEBER, THE SUCCESS OF OPEN SOURCE 38 (2004) (noting that "the very success of the proprietary paradigm increased the demand for alternatives").

38 *See* Microsoft Corp., Shared Source Initiative Frequently Asked Questions, http://www.microsoft.com/resources/sharedsource/initiative/FAQ.mspx (last visited Sept. 16, 2006) (noting that Microsoft does not want its Shared Source Initiative to be confused with "open sourcing").

39 *See, e.g.*, Eclipse, Eclipse Public License (EPL) Frequently Asked Questions, nos. 9 & 10, http://www.eclipse.org/legal/eplfaq.php (last visited June 10, 2006) (discussing business and technical advantages to open source software development).

within a larger community whose practices and norms spring from the FOSS license.[40]

Due to a variety of reasons, FOSS licenses carry indirect voice. Unlike proprietary software end user license agreements (EULAs), FOSS licenses often receive a lot of attention. It is generally understood that even as we click "I Accept" to agree to the terms, very few people read most mass market software and web site EULAs.[41] FOSS licenses, on the other hand, often engender significant debate, especially if they attempt certification to comply with the Open Source Definition (OSD).[42] The OSD certification comes from an organization in the open source camp within FOSS.[43] It certifies licenses as "open source" against a set of defined criteria. Thus, there is a chance to debate each license running that gauntlet. These debates often cover the relative merit of a license, and to what extent its terms adhere to the tenets of open source. The free software camp has a similar indirect voice mechanism for licenses. Its organization, the Free Software Foundation (FSF), maintains a web site that evaluates whether other FOSS licenses are compatible with the GPL.[44]

The most well-known FOSS license is the GPL. This license is peppered with indirect voice. It extols the virtues and goals of free software. The GPL is the most widely adopted FOSS license,[45] and at seven pages is relatively short compared to many proprietary

---

40 McGowan, *supra* note 7, at 242-43.

41 *See* Lydia Pallas Loren, *Slaying the Leather-Winged Demons in the Night: Reforming Copyright Owner Contracting with Clickwrap Misuse*, 30 OHIO N.U. L. REV. 495, 496-97 (2004) (noting that many contract terms contain "outlandish" provisions, relying on the fact that many users will not read the terms).

42 *See* Open Source Initiative, The Open Source Definition, http://www.opensource.org/docs/definition.php (last visited Sept. 16, 2006) [hereinafter OSD]. OSI is a nonprofit "corporation . . . certification mark and program." Open Source Initiative Home Page, http://www.opensource.org (last visited Sept. 16, 2006) [hereinafter OSI].

43 *See* OSI, *supra* note 42 (noting that the organization is "a non-profit corporation dedicated to managing and promoting the Open Source Definition for the good of the community").

44 Free Software Foundation, Licenses, Various Licenses and Comments About Them, http://www.fsf.org/licensing/licenses (last visited June 10, 2006) (providing a detailed explanation of the FSF's free software classification criteria).

45 Peter Galli, *GPL 3 to Take On IP, Patents*, EWEEK, Nov. 22, 2004, http://www.eweek.com/article2/0,1759,1730102,00.asp (noting that the GPL is "the most widely used free-software license").

EULAs. Moreover, Stallman wrote the GPL using the language of software developers,[46] increasing its voice-carrying capability.

Even for users who do not read licenses, using FOSS can create indirect voice, especially if the user declares that she does not believe in using proprietary software because it prohibits sharing.[47] The public aura around FOSS is that it springs from a different ideology. If a user of proprietary software declares that she will henceforth use FOSS, this may make an impression. The non-adopting user may conclude that the FOSS-adopting user has switched for the perceived lower software cost, but may also conclude that the switch is motivated by its value as a social statement.

FOSS voice, while springing from the licenses and use of the code, goes beyond such licenses and use. Part III concerns exit and voice through technologists who also contribute to FOSS projects as an extracurricular activity apart from their regular employment. This is not direct exit because the technologists have not left their employers who use or sell proprietary software. They are simply choosing to spend their nonwork time at an activity that parallels their regular job by moonlighting on FOSS. In this case, the employer is not losing customers, the paradigmatic exit in Hirschman's framework.

The extracurricular FOSS moonlighting is a mixture of exit and voice. It is exit in the sense that it might divert some focus from the technologist's regular employment, especially if the technologist is a programmer, although there could be benefits for the regular employer through training effects or other consequences. It is voice because identification with the values of FOSS may be one of the

---

46 *See* Robert W. Gomulkiewicz, *General Public License 3.0: Hacking the Free Software Movements Constitution*, 42 HOUS. L. REV. 1015, 1032, 1035-36 (2005) (explaining the way in which the FOSS community interprets the GPL, and arguing generally for a clarification of the language used in newer versions).

47 Although not my focus, FOSS code can transmit voice beyond the FOSS licenses. *See* David McGowan, *From Social Friction to Social Meaning: What Expressive Uses of Code Tell Us About Free Speech*, 64 OHIO ST. L.J. 1515, 1520-24 (2003) (arguing that some, but not all, source code is "a form of expression for purposes of the First Amendment"). Furthermore, code contains comments, which are nonfunctional statements the computer ignores. They are for other programmers. Their purpose, typically, is technical documentation, but they can be styled to promote FOSS principles. A proprietary software developer studying FOSS source code, even if she never reads the GPL, may still come to understand that the FOSS programmers have a different conception about rights in software, software sharing, and the best way to build good software.

reasons for working on the FOSS project. The literature suggests various reasons why FOSS contributors make the effort. Non-voice reasons exist, such as career advancement, where the reputation earned or skills learned on the FOSS project provide future career opportunity.[48] But identification with the values or community of FOSS is often part of the contributor's story.[49] Such identification has voice-carrying potential. It might be direct voice if the programmer broadcasts the fact of her extracurricular activities, and due to her interest in FOSS she seeks to persuade management to use more FOSS, or, if the company is a software provider, to release code as FOSS (which may also require transitioning the company to a different business model).

Beyond exit and voice, Hirschman's framework includes loyalty, which arises most plausibly when exit is minimally effective or unavailable and voice has noticeable impact.[50] Loyalty, while related to the other two mechanisms, is the most amorphous of the three mechanisms in Hirschman's framework. It is a broad rubric: someone for some reason stays and provides feedback.[51] In the job satisfaction context, scholars have extended Hirschman's framework by adding a fourth element: neglect, as an alternative to loyalty.[52] In both neglect and loyalty, the employee remains with her employer. But, in a state of neglect, the employee gives less than her best effort

---

[48] *See* Josh Lerner & Jean Tirole, *The Simple Economics of Open Source* 14-15 (HBS Finance Working Paper No. 00-059, 2000), *available at* http://ssrn.com/ abstract=224008 (discussing the "career concern incentive" literature, and identifying the "signaling incentive" that many open source programmers value, such as high visibility, significant impact, and readily accessible information regarding project performance).

[49] *See* E. Gabriella Coleman, *Three Ethical Moments in Debian* 2 (Sept. 15, 2005), *available at* http://ssrn.com/abstract=805287 (arguing that "[FOSS] projects are sites for a series of important ethical transformations"). *See also* Dan M. Kahan, *The Logic of Reciprocity: Trust, Collective Action, and Law*, 102 MICH. L. REV. 71, 71-73, 92-98 (2003) (using a theoretical framework designed to counter the central tenets of Mancur Olson's book, *The Logic of Collective Action*, to argue that individuals will contribute to goods that benefit a group to which they belong, and using the example of open source software, which has the "same individual motivations that generate reciprocal intellectual production within both the university and commercial firms that emulate the university model" to establish his premise).

[50] HIRSCHMAN, *supra* note 1, at 34, 77-78, 80, 83.

[51] *See* Laver, *supra* note 31, at 471, 477-81 (noting the difficulties with cabining loyalty, and that the exit and voice frameworks compress two dichotomies into one: "These two choices are those between Exit and Stay and between Voice and Silence").

[52] Caryl E. Rusbult et al., *Impact of Exchange Variables on Exit, Voice, Loyalty and Neglect: An Integrative Model of Responses to Declining Job Satisfaction*, 31 ACAD. MGMT. J. 599, 601 (1988).

or her job satisfaction is less than that in the condition of loyalty. The extracurricular FOSS contributor can be characterized to fit this extended version of the basic framework: the dissatisfied technologist seeks indirect exit from the state of neglect by after-hours contributions to FOSS projects.

Moving beyond the extracurricular FOSS contributor, Part IV describes how both the free software and open software camps engage in general activism through advocacy, license enforcement, and lobbying. Groups in both FOSS camps evangelize FOSS in their own ways. For the free software camp, Stallman's self-proclaimed most important role is no longer to program FOSS but to evangelize the free software philosophy.[53] The open source camp has Eric Raymond, whose writings about FOSS are well-known, and who also travels and speaks about FOSS.[54] The open source camp also has many corporate representatives. These are individuals who are employed by companies such as IBM or Red Hat with the responsibility of interfacing with the various FOSS subcommunities.[55] This corporate activism for FOSS tends to emphasize the open source camp's approach.

Beyond general activism, both camps actively enforce FOSS licenses and lobby government. The enforcement actions function as a form of FOSS advocacy, thereby carrying indirect voice. Thus far, they have rarely resulted in litigation in the courts. The FSF has been active in GPL license enforcement. An affiliate of the FSF, Professor Eben Moglen of Columbia Law School and general counsel of the FSF, was involved in a GPL enforcement action that produced one of the few United States court cases mentioning the GPL.[56] The enforcement actions generally target companies who are

---

53 Richard Stallman, Free Software: Freedom and Cooperation, Presentation to the University of Pittsburgh ACM Chapter (Apr. 7, 2005) (author's notes of presentation on file with author).

54 *See* Eric S. Raymond, Eric S. Raymond's Home Page, http://www.catb.org/~esr (last visited July 4, 2006).

55 *See, e.g.*, Open Source Business Conference, Biography for Stephen Mutkoski, http://www.idgworldexpo.com/live/13/events/13SFO06A/conference/bio//CMONYA00BDZ4 (last visited June 10, 2006) (describing Mutkoski as a senior attorney with Microsoft Corporation whose responsibilities include a variety of external interfacing activities with the open source community).

56 *See* Declaration of Eben Moglen in Support of Defendant's Motion for a Preliminary Injunction on Its Counterclaims at 3-9, 11, Progress Software Corp. v. MySQL AB, 195 F. Supp. 2d 328 (D. Mass. 2002), *available at* http://www.gnu.org/press/mysql-affidavit.pdf [hereinafter Moglen Declaration].

using GPL protected software and who have not provided the source code or who are otherwise violating the GPL's software freedom conditions.[57] They often receive coverage by the specialized press, a potential voice channel in Hirschman's framework.[58] Additionally, some FOSS activism blossoms into lobbying. In Europe, for example, FOSS groups were influential in lobbying the European Union Parliament against a proposal related to software patents.[59]

Each aspect of exit and voice catalogued in Parts I through IV reflect the passionate politics and unique economics characterizing FOSS. User exit from proprietary software triggers the voice embedded in the license. Extracurricular FOSS contributors can quietly protest by working on projects on their own time, while FOSS advocates actively marshal the movement's voice in a variety of ways. These intertwined and reinforcing mechanisms are an important part of the FOSS story and should be a part of the legal and policy considerations channeling its future.

## I
## EXIT FROM PROPRIETARY SOFTWARE TO FOSS

Many FOSS users switched from some proprietary-licensed software application to a FOSS equivalent, or chose between these two for a new application. Before the advent of FOSS licensing, virtually every computer user ran some proprietary-licensed software. Users had few options to exit the traditional approach to licensing software, and none of the options represented the paradigm shift offered by FOSS.

---

[57] Free Software Foundation, Violations of the GPL, LGPL, and GFDL, http://www.fsf.org/licensing/licenses/gpl-violation.html (last visited June 10, 2006) (providing a checklist of potential GPL violations, including whether source code is included in the distribution or not).

[58] *See* HIRSCHMAN, *supra* note 1, at 4. Moreover, the voice FOSS receives may be artificially amplified in the press due to the novel characteristics of FOSS licensing. Peter Holditch, *Measuring the Value of Software Infrastructure: What Do You Get for Your License Fee?*, WEBLOGIC DEVELOPER'S J., Feb. 11, 2005, *available at* http://wldj.sys-con.com/read/48218.htm (reporting one developer's view that FOSS received more attention than warranted for enterprise software applications due to "the media's love of controversy").

[59] *EU Rejects Controversial Software Patents Proposal*, EWEEK, July 6, 2005, *available at* 2005 WLNR 10689609 (noting that "open-source leaders such as Linus Torvalds have spoken out against the . . . [European Union's computer-implemented inventions] directive").

This Part takes as a given that FOSS offers a unique alternative to traditionally licensed software. Along with describing the benefits of typical FOSS licenses, it sketches the characteristics of FOSS licensing that might cause concern among users. Some of these characteristics invite uncertainty due to their novel nature and may omit allegedly beneficial provisions expected in traditional software licenses. In this context, this Part explores the pros and cons of the FOSS exit opportunity for software users, including the voice that results from a user's exit decision and how that voice might reinforce exit.

### *A. The Nature of FOSS Exit*

Exit to FOSS manifests in a number of ways. First, a user might replace an existing application. This occurs, for example, when a user replaces a Unix or Windows computer with a GNU/Linux computer. Second, a user might decide whether to use proprietary software or FOSS in a new application. This second exit opportunity often occurs with Internet applications. During the initial growth period of the Internet in the late 1990s and early 2000s, both types of applications were available for key Internet software components. FOSS applications captured much of the Internet infrastructure market as companies took the "exit" option by running FOSS rather than proprietary-licensed software for these new applications.[60] Most such FOSS adopters, however, also ran proprietary software in the legacy portions of their IT infrastructure.

Most FOSS users run both types of software because FOSS equivalents to traditional software are only available in a small, but increasing, number of application categories. For example, the market for desktop operating systems is a highly visible market but currently has minimal FOSS penetration.[61] Most organizations that deploy some FOSS have many more computers running Microsoft's Windows operating system than computers running a FOSS operating system. This is because GNU/Linux, the primary FOSS operating system, currently is not generally perceived by the marketplace as equivalent to Windows for desktop users.[62] To

---

[60] *See* GLYN MOODY, REBEL CODE: INSIDE LINUX AND THE OPEN SOURCE REVOLUTION 19, 182 (2001).

[61] On the other hand, the desktop represents a large FOSS exit opportunity, restrained mainly by compatibility issues. *See* Bulkeley, *supra* note 21.

[62] *See generally id.*

overgeneralize, a necessary part of generating an "equivalent" application for desktop users is to emulate the user familiarity of Windows and perceived ease-of-use for its interface. Because GNU/Linux allegedly does not provide equivalent user familiarity or ease-of-use, the average nontechnical computer user is often discouraged from adopting the application. Without the perception that the features and functions are equivalent, exit from Windows to GNU/Linux for the mass of desktop users has been muted in comparison to the exit in other application classes. The need for equivalent (or better) functionality is typically a necessary, but not sufficient, condition for exit to FOSS.

In response to this need, the number of FOSS-equivalent applications has grown over time. Precisely why the FOSS movement responds to fill this need is a very complex and interesting topic, but one beyond the scope of this Article. The fact remains that FOSS alternatives have allowed many users to exit proprietary software in certain parts of their IT infrastructure but not in others. This raises a second question of equivalent complexity: which application types are more suitable or inclined to FOSS development?[63] This question is also not my focus, but as with the first question, it is relevant to the exit opportunities. FOSS programmers may generate certain application types at a greater rate than others for motivations not necessarily fully understood.[64] If the projects are successful, they may eventually generate functionality equivalent to or surpassing proprietary software.[65] This creates an

---

[63] Eric S. Raymond, The Magic Cauldron, § 10 (1999), http://catb.org/~esr/writings/magic-cauldron/magic-cauldron.html (discussing conditions that may determine when it is beneficial for a software application to be open or closed source).

[64] As technology experts, FOSS project leaders may have an intuitive feel for the types of applications that have the best chance of gaining a user base commensurate with the developer's goals. *See generally* WEBER, *supra* note 37, at 11-12 (describing the political economy inquiries raised by open source, including the coordination question: how do FOSS contributors choose what projects to work on and coordinate within the project?).

[65] The FOSS functionality might be better than the functionality of proprietary software, but my analysis assumes that it is sufficiently equivalent to be a substitute. *See generally* James W. Paulson et al., *An Empirical Study of Open-Source and Closed-Source Software Products*, 30 IEEE TRANSACTIONS ON SOFTWARE ENGINEERING 246, 254-55 (2004) (finding empirical support for only some of the common beliefs about the differences between FOSS and proprietary software, namely the notions that creativity is more prevalent in FOSS, and that defects are discovered and repaired more quickly in FOSS).

exit option where the switching decision primarily hinges on the licensing differences and the user's switching costs.

As equivalent FOSS applications pop up, there are typically some early adopters using these applications. In the case of FOSS, the software technology is often not the revolutionary aspect of the FOSS value proposition. Rather, the innovation is in the licensing terms. Some commentators have suggested viewing informational assets such as software as a bundle of benefits, where the software functionality is commingled with the licensing terms in the user's evaluation.[66] Even this view applies in my analysis because I intend to examine the case where the software functionality of FOSS and proprietary licensed software is roughly equivalent.

In Hirschman's *Exit, Voice, and Loyalty* framework, users choose the exit opportunity when the quality of the incumbent firm's product declines beyond their tolerance.[67] The subtitle to the book illustrates this, declaring that the framework is designed to help understand "Responses to Decline in Firms, Organizations, and States."[68] My analysis characterizes the quality gap as the

---

66 *See* Robert W. Gomulkiewicz, *The License Is the Product: Comments on the Promise of Article 2B for Software and Information Licensing*, 13 BERKELEY TECH. L.J. 891, 896, 899 (1998) (discussing how mass market licenses provide software users with a variety of rights, sometimes more than the "user would have acquired had the user simply bought a copy of the software, including reproduction, derivative works, and distribution rights").

67 *See* HIRSCHMAN, *supra* note 1, at 4, 24-25, 36, 47-49. From this basic observation about what triggers exit, Hirschman evaluates a range of situational and structural factors that influence the availability and effectiveness of exit as a force that disciplines a firm from a recoverable lapse in quality. *See id.* at 4, 24-25, 34-43. One of Hirschman's fundamental points is that, while the standard economic model is perfect competition, that condition is not predominate in competitive markets. *Id.* at 21-25. Exit is the paradigmatic, optimal choice for competitive markets. This observation, however, is Hirschman's take-off point; since so many markets are not perfectly competitive, exit is comparatively less effective and this opens the analysis to consideration of the voice mechanism. *See id.* at 25-27, 27 n.7, 29. Scenarios Hirschman discusses include the case where products are highly differentiated, or where customer preferences are highly attuned to product particulars. *See, e.g.*, *id.* at 48-52. Both descriptions are often applicable to software products, particularly to "back-office" IT infrastructure software. With a wide variety of available products, quality may have to decline much more than in a competitive market before a customer will switch. *See id*. With software, this often occurs due to the dominant effects of switching costs or network effects. The quality preference of Hirschman's framework is, in part, the lock-in effect traditionally discussed in reference to software.

68 Hirschman notes that exit and voice may work better in tandem when a firm's customers have a range of quality elasticity preferences and a differing proclivity to notice a quality decline, that is, when there are some alert and inert customers. *See id.* at 24, 32, 48, 63-64. *See also* Laver, *supra* note 31, at 465 (discussing generally the

differences between traditional licensing and FOSS licensing.[69] I put aside significant software functionality differences by assuming an equivalent FOSS offering.

Using this assumption, I analyze three separate categories of users: free software advocates, open software advocates, and corporate users.[70] For each of these three groups, I inquire: (1) in what sense would the user understand "quality decline" in software licensing; (2) what key FOSS licensing terms produce the attraction that makes FOSS an attractive exit; and (3) what role might voice play in that exit? This inquiry helps explore the role of exit and voice in user FOSS adoption, but the questions it poses cannot be answered in an empirically verifiable way. While I will suggest my sense of the issues posed by these inquiries, the questions themselves, and the structural points they highlight, are more important.

### *1. Exit for Free Software Advocates*

As the first and foremost free software advocate, Richard Stallman invented free software through his rebellion against proprietary-licensed software. To free software advocates, "quality decline" in software began a long time ago when proprietary

---

different ways in which consumers react to changes in a product's quality). If too many customers exit at once, the firm cannot recover, but if a few exit (enough to be noticeable by management), complemented by alert and inert customers giving voice, a dual disciplining effect is created. *See* HIRSCHMAN, *supra* note 1, at 24, 38. Moreover, the tendency to give voice is increased when dealing with costly or durable goods. *Id.* at 40-41. Enterprise and platform software applications are usually classified as differentiated and durable goods.

69 Saying that a quality "gap" arises from licensing differences sidesteps the question of defining quality. Hirschman notes that there is an alternative method to specify product quality variations: calculate a "price equivalent." *See* HIRSCHMAN, *supra* note 1, at 48. But for purposes of this Article, following Hirschman's treatment, quality is conceptualized as a rough rubric with both subjective and objective elements. *See id.* at 50-53, 54 & n.8, 141-43, 144 & n.4, 145. *See also* Helen Hershkoff & Adam S. Cohen, *School Choice and the Lessons of Choctaw County*, 10 YALE L. & POL'Y REV. 1, 23 (1992) (reflecting on Hirschman's theory of consumer behavior as it relates to product quality).

70 I exclude one large class of users from the taxonomy: individual nontechnical users. In the three groups I taxonomize, technologists are assumed to occupy the relevant authority positions. Some of the discussion for these three groups will apply to nontechnical individuals, but their grouping in the user population does not provide a clear organizational vehicle for the analysis, especially since most FOSS use is under the purview of technologists. Moreover, nontechnical users, to the extent they use FOSS, are more likely to have done so under decision processes that are not strategic in the sense that this Article seeks to explore.

licensing became the commercial norm.[71] The commercial approach often keeps the source code secret, charges for use, and licenses only a defined field, type, or range of use. Moreover, further development or self-help is not available or feasible. Without the source code, a user cannot readily modify the code or have someone other than the licensor modify it. All these proprietary software characteristics were antithetical to quality for free software advocates. Therefore, the FOSS license that started the movement, the GPL, is inapposite to each proprietary characteristic.[72]

The GPL's terms are a quality-indictment of proprietary software. For this group of users, software should come with source code and be unhindered by royalty charges as to its use.[73] Moreover, the source code should continue to be open and free as it evolves through future development.

This last point explains much of the rest of the GPL. The license uses the rights of copyright to implement a set of conditions attempting to ensure permanence for source code availability and anti-royalty provisions, as well as to resist other threats to the FOSS paradigm. Upon a distribution of the software, the GPL requires that a distributor provide the source code, not charge royalties, and reapply the GPL's terms to downstream licensees for the original code and other software sufficiently intermingled with the original code.[74] The full detail of these conditions is not critical to the analysis here. They generally provide the effect sought: a mode of software licensing that preserves the code's form to the preferences of the free software advocates.[75]

---

71 *See* MOODY, *supra* note 60, at 19, 26-29 (describing the steps taken by Richard Stallman initially to develop the GNU project and the GPL).

72 *Id.* at 26-29 (quoting Stallman as saying, "If I had been developing proprietary software, I would have been spending my life building walls to imprison people").

73 Free Software Foundation, The Free Software Definition, http://www.fsf.org/licensing/essays/free-sw.html (last visited June 10, 2006) (defining free software by delineating the four kinds of freedom necessary for the software to be free as "a matter of liberty": (1) "freedom to run the program"; (2) "freedom to study how the program works, and adapt it to your needs"; (3) "freedom to redistribute copies so you can help your neighbor"; and (4) "freedom to improve the program, and release your improvements to the public, so that the whole community benefits").

74 GPL, *supra* note 7, at 1-3, 6.

75 This Article accepts as a premise, without claiming that it is proven, that the licensing system works well enough to provide the exit opportunity. However, there are various issues of doctrine that are not necessarily well settled within FOSS licensing. *See* McGowan, *supra* note 7, at 289-302 (discussing doctrinal questions related to a variety of issues, including assent, privity, term, termination, and assignment).

The GPL's substantive license terms fashion the exit opportunity Stallman desired. In addition, the license advertises the purpose, philosophy, and nature of that exit. The success of its substantive terms may be equaled or surpassed by its precatory language expressing the indirect voice of the FOSS movement. It has been labeled by third parties and the FSF as the "constitution" of the FOSS movement.[76] Any number of quotes from the GPL would demonstrate its constitutional, indirect voice-carrying character, but the first two sentences of the preamble will suffice:

> The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.[77]

In Hirschman's framework, the GPL is an institutional arrangement that mixes exit and voice, enabling both to operate on users of proprietary software.[78] Early in his book, Hirschman conceptualizes voice in a narrow way: the message of current customers who complain about quality decline but have not yet switched.[79] Since most FOSS users also use proprietary software, under this narrow definition of voice their exit in one application class will likely generate voice to their proprietary software providers in other application classes through communications typical of the ongoing vendor-to-customer relationship.[80]

---

[76] Rod Dixon, *Breaking into Locked Rooms to Access Computer Source Code: Does the DMCA Violate a Constitutional Mandate When Technological Barriers of Access Are Applied to Software?*, 8 VA. J.L. & TECH. 2, ¶ 106 n.257 (2003), http://www.vjolt.net/vol8/issue1/v8i1_a02-Dixon.pdf; Li-Cheng (Andy) Tai, The History of the GPL (July 4, 2001), http://www.free-soft.org/gpl_history; Richard Stallman & Eben Moglen, GPL Version 3: Background to Adoption, http://www.fsf.org/news/gpl3.html (last visited Sept. 16, 2006).

[77] GPL, *supra* note 7, at Preamble.

[78] *See* HIRSCHMAN, *supra* note 1, at 33-34, 124 (illustrating the "see-saw relationship between exit and voice" by pointing to the many citizen complaints about quality and services in Soviet Russia). FOSS licenses generally enable exit from proprietary software in ways analogous to Tim Wu's application of peer-to-peer file sharing (i.e., enabling certain groups to organize and thus avoid copyright law). *See* Wu, *supra* note 29, at 697-709. In Wu's application, the code is software; in my analysis, the code is legal code in the FOSS license. Moreover, the FOSS license provides coordination benefits that enable FOSS users and developers to coalesce around the software with a web of interlocking incentives that to some degree limit collective action problems.

[79] *See* HIRSCHMAN, *supra* note 1, at 4.

[80] The relational aspects of software vendor-to-customer engagements can be quite entangled. See Franklin G. Snyder et al., *Relational Contracting in a Digital Age*, 11

Hirschman later broadens his concept of voice to include the signal that occurs upon customer exit and general activism by former customers.[81] In this latter voice characterization, which I call indirect voice, the GPL seems a particularly important vehicle.

In a noisy, voice-laden way, the GPL defines the exit alternative the free software advocates desire. It provides software with relatively unfettered functional freedom so that users can tinker with, or exercise full control over, the source code if they wish. The terms also make it relatively easy for a user to have others modify the software for her. The license is universally available via the Internet through its posting, with commentary, on the FSF's web site.[82] Its preamble and other provisions express the voice that springs from that group's values, politics, and preferences.[83]

The indirect voice expressed in the GPL also recruits other users to adopt FOSS. This makes the exit more attractive in both the political and economic sense. As like-minded FOSS users build their community, they enjoy the social satisfaction of solidarity with an increasingly numerous group. Due to the effects of network economies, an increasing user base infuses greater value into the software and the FOSS licensing method. Considering these diverse effects, the role of indirect voice in the exit process of free software advocates would seem to be substantial, perhaps even predominant in the sense that the values and ambitions in the free software message are surely a critical part of engendering the code contributions that launched the FOSS movement in the first place.[84] Parts of the FOSS phenomenon have taken their own trajectory, but there is no doubt that the free software advocates are the

---

TEX. WESLEYAN L. REV. 675 (2005) for a discussion suggesting analysis using relational contracts literature. This panel also examines, in parallel, relational contract theory in light of the "vast changes wrought by the information revolution." *Id.* at 678. Enterprise software typically provides a business-critical function. Most users purchase ongoing maintenance and support, and sometimes a base software product or technology is modified to suit a customer's unique needs. It is common for enterprise software suppliers to have personnel in the customer's facility or a remote presence on the customer's computers. These entanglements provide ample opportunity for the user to exercise voice with the supplier, and in particular to show the proprietary software supplier that the customer is testing FOSS in some part of its operation.

81 *See* HIRSCHMAN, *supra* note 1, at 22-25, 35 n.7, 37-38.

82 Free Software Foundation, *supra* note 44.

83 *See* GPL, *supra* note 7, at Preamble.

84 *See* MOODY, *supra* note 60, at 26-30.

fountainhead of the licensing exit device and the indirect voice that amplifies it.[85]

### *2. Exit for Open Software Advocates*

The line between open source software advocates and free software advocates is not bright. Although the categorization is somewhat tenuous, I analyze each group separately to highlight the open source group's emphasis on exit as compared to voice. Both mechanisms are important for both groups, but open source advocates have a focus that emphasizes exit—i.e., open source development as a superior way to generate superior software. Linus Torvalds, the leader of the Linux kernel FOSS project, exemplifies this pragmatic thrust. The work of the Open Source Initiative (OSI) also seems in this vein, as its web site describes:

> The basic idea behind open source is very simple: [w]hen programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.[86]

Since FOSS exit generally is more likely when equivalent or better FOSS alternatives exist, the open source advocates' role has been to promote licensing terms that facilitate the process of producing superior software. Some of this work occurs through the Open Source Definition (OSD), which defines criteria against which the OSI evaluates and certifies licenses. The OSD license criteria share many similarities with the GPL. The OSI categorizes the GPL as an "open source" license. One difference, however, is that unlike the GPL, the OSD does not require that a license demand that modifications be distributed under the same terms. The OSD merely says that a license must allow such a condition, but a license need not have it.[87] In Hirschman's framework, this difference means that an OSI-compliant license that does not demand reapplication of its terms has less voice-carrying potential because it will not necessarily propagate along with the code. Another license with nonconflicting substantive terms will suffice. This is a structural observation, not

---

[85] *See id.*

[86] OSI, *supra* note 42.

[87] OSD, *supra* note 42 ("The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.").

an empirical one. It may be that most licenses require reapplication of their terms like the GPL, but the OSD's relaxation on this point indicates that it deemphasizes voice in comparison to exit.

### *3. Exit for Corporate Users*

Together, both free software advocates and open source software advocates have driven a movement that benefits many corporate software users. While a marketplace debate is ongoing about the cost of ownership for FOSS versus proprietary software, many reports show increasing FOSS use by corporate IT departments.[88] Their FOSS adoption is perhaps the most important exit inquiry in this section.

To clarify the taxonomy, by corporate users I mean companies who use FOSS in their operations, not companies such as IBM who have invested in the FOSS movement in order to sell complementary products and services. I count companies such as IBM and Red Hat in the taxonomy as open source advocates. They are instrumental in furthering the exit opportunity FOSS provides to a much wider class of users: their corporate IT department customers.

---

[88] *See, e.g.*, Martin Butler, *Hidden Costs of Open Source*, IT WEEK, July 21, 2004, http://www.itweek.co.uk/itweek/comment/2086191/hidden-costs-open-source (speculating that open source has "been hijacked by commercial enterprises" and users should investigate its true costs); Steve Hamm, *Linux Inc.; Linus Torvalds Once Led a Ragtag Band of Software Geeks. Not Anymore. Here's an Inside Look at How the Unusual Linux Business Model Increasingly Threatens Microsoft*, BUS. WK., Jan. 31, 2005, at 60 (arguing that Linux is a more affordable option than the proprietary software Windows).

Various reports reveal mixed FOSS and GNU/Linux growth rates. While some show a continued acceleration, others show the acceleration slowing down. *See, e.g.*, Charles Ferguson, *How Linux Could Overthrow Microsoft: The Open-Source Movement Is the Largest Threat the Software Giant Has Ever Faced. Does Bill Gates Have a Plan*?, TECH. REV., June 2005, at 69, *available at* 2005 WLNR 8789992 (discussing IDC surveys that indicate revenues from Linux servers are growing at more than 40% annually, whereas server revenues for Windows are growing at less than 20% per year); Hamm, *supra*, at 60 (referring to a Forrester Research, Inc. survey that indicated that 52% of business users are switching from Windows to Linux servers); Jennifer Mears & Ann Bednarz, *Branching Out: Comfortable with Linux, Organizations Look for Opportunities to Employ Open Source Tools*, NETWORK WORLD, July 4, 2005, at 15, *available at* 2005 WLNR 10973666 (citing a Forrester Research, Inc. survey of 128 information technology decision makers that revealed nearly 75% use open source or Linux now, or plan to within the next year); Darryl K. Taft, *Slew of Fears Slow Open-Source Uptake*, EWEEK, Jan. 25, 2005, http://www.eweek.com/article2/0,1759,1753474,00.asp (discussing a SourceLabs, Inc. study that attributes the slow adoption of open source beyond Linux to customers' concerns over support and maintenance).

FOSS presents the corporate user with a set of pros and cons that is unique and unprecedented in the history of computing, making FOSS a new form of exit from proprietary software applications. Switching to FOSS may create some costs for corporate users. However, FOSS may overcome those costs because its most popular licenses require a use-price of zero. FOSS may require greater self-reliance and technical savvy, but there are no prohibitions on hiring a consultant to revise or optimize the software.

There is sometimes misunderstanding about what a corporate IT user must or must not do with such modifications. Unless the corporate user distributes the software, in the copyright meaning of that term, the FOSS license typically does not require adherence to the full set of conditions. The GPL uses this approach, meaning that a company can revise and optimize FOSS for internal use without making the source code for those changes available. Corporate users may view this as a positive. It lowers their cost of exit to FOSS because they are not forced to expend resources on contributing code back to the community as long as they do not distribute it.[89] Contrasted with this positive, an opportunity cost for using FOSS exists, as modifications cannot typically be privatized for incorporation into a commercially licensed product. Thus, corporate users may have to partition and segregate software and their IT infrastructure to avoid intermingling FOSS and other software, if they want to preserve future opportunities to externally commercialize the other software. This license management and software tracking is already a part of many IT department procedures.[90]

Given the pros and cons associated with FOSS, the anti-royalty license term might dominate at the time of an exit decision for corporate users, especially since companies are always interested in operational cost reduction. On the other hand, the source code availability term might dominate if the exit motivation is to escape

---

[89] Conditions of a FOSS license in use under the GPL typically apply only if the FOSS is distributed outside the company. For example, changes Google makes to its operational open source software are not available to the general public. *See Google's Summer of Code Pays Students to Do Open Source*, DATAMONITOR, June 9, 2005, *available at* 2005 WLNR 9127797.

[90] Cary H. Sherman & David M. Hornik, *How to Avoid the Software Police and What to Do When They Knock on Your Door*, 369 PRACTISING L. INST. 495, 534-37 (describing software asset management and tracking programs and alternatives).

the control a proprietary software vendor wields over its licensees.[91] In addition to these two questions, there is the question of voice. Is the exit motivated in response to indirect voice from the FOSS community? Is the exit designed to provide direct voice to the corporate user's proprietary software providers? Does the exit have the voice effect of making future exit more credible, as a threat, and as a disciplining force on the proprietary software vendors? All of these questions express a possible role for voice in the corporate user's view of FOSS alternatives. The next two sections explore these questions further.

### *B. Influences That May Chill Exit to FOSS*

While FOSS licenses offer advantages, they also omit standard proprietary license provisions touted as beneficial. The two foremost examples are warranties and indemnification.

Warranties for most mass-market software products typically provide minimal benefit. For example, the software might be warranted to be in accord with its manuals or a general description, and the warranty often only provides for return of the purchase price in the event of breach.[92] Warranties for high-end software products costing tens or hundreds of thousands of dollars are often much more substantial. These are sometimes negotiated in the procurement transaction and can provide important protections for corporate users. In contrast, as a result of the FOSS-distributed community development model, most FOSS licenses offer no warranty.

Traditional software licenses often indemnify the licensee if a claim of intellectual property infringement is brought against the

---

[91] *See* Ferguson, *supra* note 88 (stating that proprietary software locks in its users so they become a "hostage to the software vendors whose products they buy").

[92] *See, e.g.*, EMC Software, Documentum Software License Agreement, http://software.emc.com/about_us/legal/Documentum_Software_License.pdf (last visited June 10, 2006) ("EMC warrants that the Software will perform substantially in accordance with the Documentation for the ninety (90) day period following shipment of the Software when used on the recommended operating system and hardware configuration and in accordance with the Documentation. Non-substantial variations of performance from the Documentation does not establish a warranty right. Any claims submitted under this section must be submitted in writing to EMC within the specified warranty period. EMC's sole and exclusive obligation for warranty claims shall be to make the Software operate as warranted or, if EMC is unable to do so, to terminate the license for such Software and return the applicable license fees paid to EMC for the applicable Software.").

licensee for her use of the product.[93] These protections are important for corporate users. Most FOSS licenses do not contain an indemnity provision. Only in the few years before this writing did any indemnification options appear for FOSS.[94] Along with warranty protection, indemnification was something corporate users would typically not expect to receive with FOSS. The question this raises is to what degree does omission of these two protections diminish a corporate user's taste for FOSS?

Another potential chilling effect for FOSS generally, and specifically for its flagship, GNU/Linux, is the SCO litigation. The details of this situation are well documented elsewhere,[95] so I recount them here only briefly. SCO's predecessor licensed source code to IBM. In a case filed in March 2003, SCO claimed that IBM contributed some of that code to the Linux kernel, thereby violating the original license contract and its trade secret provisions.[96] If true,

---

93 Typically, indemnification clauses in enterprise or corporate proprietary licensed software are applied to the product or software in unmodified form, and do not cover use of the product or software in a greater system if that greater system infringes a patent. Sometimes corporate users negotiate for the rights to some or all of the source code in a product. Modifying the original could create a patent-infringing technology even if the product was infringement-free as originally shipped. *See* discussion *infra* Part IV.C.

94 For example, FOSS distributors HP, SuSE, and JBoss only provide indemnification to their customers under certain specific circumstances. *See* Phil Hochmuth, *HP to Linux Users: We Got Your Back. But Does It Really*?, NETWORK WORLD, Oct. 1, 2003, http://www.networkworld.com/newsletters/ linux/2003/0929linux2.html (discussing HP's commitment to indemnify certain Linux customers); HP, Open Source and Linux from HP, https://h30201.www3.hp.com/default.asp (last visited July 12, 2006) (providing information and the opportunity for customers to register for the indemnification process); *JBoss Enhances Indemnification Program*, EWEEK, Apr. 5, 2005, *available at* 2005 WLNR 7378344 (announcing JBoss's plans to enhance its indemnification coverage to include "unlimited coverage for defense, repair and replacements involving any intellectual property claims"); Robert McMillan, *Novell to Indemnify SuSE Customers*, NETWORK WORLD, Jan. 12, 2004, http://www.networkworld.com/news/2004/0112noveltoin.html (revealing Novell's plans to indemnify SuSE Linux Enterprise 8 customers).

95 *See, e.g.*, Nina L. Chang, Comment, *No GNU Is Good G'News for SCO: Implications of* SCO v. IBM, 9 INTELL. PROP. L. BULL. 47, 47 (2004); Kerry D. Goettsch, Recent Development, SCO Group v. IBM: *The Future of Open-Source Software*, 2003 U. ILL. J.L. TECH. & POL'Y 581, 583-84; Andrew LaFontaine, Comment, *Silicon Flatirons Student Writing Contest 2005: Adventures in Software Licensing:* SCO v. IBM *and the Future of the Open Source Model*, 4 J. ON TELECOMM. & HIGH TECH L. 449, 468-80 (2006); *see also* The SCO v. IBM Info Website, http://sco.iwethey.org (last visited July 7, 2006) (providing updates and a detailed summary of the litigation). Another site generally following the litigation is available at http://www.groklaw.com.

96 *See* Complaint at ¶ 1, Caldera Systems, Inc. v. Int'l Bus. Machs. Corp., No. 2:03cv0294 (D. Utah Mar. 6, 2003), *available at* http://sco.tuxrocks.com/Docs/IBM/

this would mean that unauthorized copies of the code would be in the hands of many users of Linux kernel based operating systems, allowing SCO to bring copyright infringement claims against such users, which it did in two cases.[97]

Ever since the SCO case began, new users continued to switch to GNU/Linux, although reports differ on whether or to what degree the case slowed or chilled user adoption.[98] Many explanations are possible for the continuing growth in use, but it is a notable phenomenon given the shadow the case casts on intellectual property rights in GNU/Linux. Users might not have known about the case, or they may have felt that the case was weak. The FOSS value proposition for GNU/Linux may have swamped any risk users felt from the SCO litigation. Another potential factor sustaining user adoption despite the case may be indirect voice, where users switch even with knowledge of the SCO-related risks based in part on their response to the FOSS message.[99] The SCO litigation related to only the Linux kernel, but it also raised questions about the intellectual property pedigree of FOSS more generally. These concerns have evoked some revised practices in the licensing and use of FOSS,[100] but they generally have not slowed down the FOSS bandwagon or its effects.

---

complaint3.06.03.html [hereinafter SCO Complaint]; Answer at ¶ 1, Caldera Systems, Inc. v. Int'l Bus. Machs. Corp., No. 2:03cv0294 (D. Utah Apr. 30, 2003), *available at* http://www.groklaw.net/pdf/Doc-13.pdf [hereinafter IBM Answer].

97 *See* David Bank, *SCO Broadens Its Attack on Linux; Suits Against AutoZone, DaimlerChrysler Claim Breach of Rights on Unix*, WALL ST. J., Mar. 4, 2004, at B5. Although both cases stem from Linux use, they are different. SCO's suit against AutoZone is in federal district court, based on copyright infringement. Complaint at 6-7, ¶¶ 20, 22, SCO Group, Inc. v. Autozone, Inc., No. CV-S-04-0237-DWH-LRL (D. Nev. Mar. 3, 2004), *available at* http://www.thescogroup.com/scoip/lawsuits/autozone/20040303_AZ_complaint.pdf. The suit against DaimlerChrysler is in Michigan state court, based on a license agreement SCO has with DaimlerChrysler. Complaint at 5, ¶ 20, SCO Group, Inc. v. DaimlerChrysler Corp., No. 04-056587-CK (Mich. Cir. Ct. Mar. 3, 2004), *available at* http://www.thescogroup.com/scoip/lawsuits/daimlerchrysler/Complaint-and-Jury-Demand-March-3,2004.pdf. SCO alleges that DaimlerChrysler is in breach for failing to provide a certification that it is not in violation of the agreement's provisions due to its use of Linux. *Id.* at 7, ¶¶ 27-28.

98 *See* Ferguson, *supra* note 88, at 69; Hamm, *supra* note 88; Mears & Bednarz, *supra* note 88, at 16; Taft, *supra* note 88; Vetter, *supra* note 7, at 643 n.231.

99 *See* Laver, *supra* note 31, at 465-67 (discussing voice in the Hirschman framework as a feedback mechanism that may operate on other users, or the general public).

100 *See Linux: Plans Are Being Adopted for Method to Track Updates*, WALL ST. J., May 25, 2004, at B6 (reporting that in light of the SCO case, Linux kernel developers would henceforth be asked to submit a "Developer's Certificate of Origin, [which] is designed to ensure the correct attribution of submissions to developers").

### *C. Disciplining Effects from Exit to FOSS*

One noticeable effect of FOSS has been the various instances of proprietary-oriented IT companies embracing FOSS in one way or another. These instances include companies releasing former proprietary products as open source products,[101] and complementary distributors such as Red Hat and IBM jumping on the bandwagon.[102] These effects do not fit directly in the Hirschman framework, which posits exit and voice as mechanisms that help a firm correct from a recoverable lapse in "quality." In my application of the framework, these examples are more akin to a firm becoming the competition—which is what happens when a proprietary product is released as FOSS.

Within the Hirschman framework, however, one can find examples of the disciplining effects of exit, particularly by looking at Microsoft's responses to FOSS generally and GNU/Linux specifically. These examples are: heightened indemnification, price discounting, Microsoft's Shared Source program, and the computer security issue.[103]

Once it understood the pros and cons of FOSS licensing, Microsoft improved its indemnification provisions to heighten customer benefits in its licenses. This occurred in two steps. First, Microsoft increased indemnification benefits for corporate end users.[104] Later, it extended these protections to its distributors.[105] Since most FOSS licenses do not provide indemnification, Microsoft's license revisions can be seen in the Hirschman framework as an improvement in the quality of the joint benefit arising from the software product and the license.

---

[101] See Hamerly & Paquin, *supra* note 15, at 197-206, for an explanation of the events leading up to, and culminating in, Netscape's decision to release its source code. *See also* Don Clark, *Sun to Share Source Codes for Some Java Programming: Software for Server Systems to Be Included in Attempt to Court Users of Linux*, WALL ST. J., June 27, 2005, at B5.

[102] *See supra* text accompanying notes 21-22.

[103] *See infra* text accompanying notes 104-13.

[104] *See* Robert A. Guth, *Microsoft Extends Legal Protections: PC Makers, Partners Get Indemnification in Effort to Combat Use of Linux*, WALL ST. J., June 23, 2005, at B4 (discussing how Microsoft will provide indemnification to distributors based on the amount of business they do with Microsoft).

[105] *See* Steve Lohr, *Microsoft Will Pay Legal Costs If Technology Partners Are Sued*, N.Y. TIMES, June 23, 2005, at C4 (discussing Microsoft's plan to extend indemnification protection to include distributors).

Indemnification, however, is a legal protection that is only valuable to the customer if she is sued. As such, customers may not value it at the time of product evaluation with the same degree of intensity as other terms, such as price. A commonly perceived advantage of FOSS is superior price.[106] For corporate users, the price of FOSS is not necessarily zero because most FOSS licenses allow a distributor to charge for ancillary services, including distribution costs, aggregation and bundling, ongoing support, updates, and even additional legal protections.[107] Thus, while GNU/Linux is available for download on the Internet literally free of charge, many companies pay Red Hat subscription fees for a package of services related to a copy of Red Hat's distribution of GNU/Linux. However, most FOSS distributors can generally undercharge their proprietary software competitors. This reality has led to a propaganda battle concerning whether FOSS has a lower cost of ownership, assuming that more internal company resources are required to operate and manage FOSS. Given the FOSS price advantage, one response reported in some sales situations is deep price discounting by Microsoft as an attempt to dissuade international customers from exiting to GNU/Linux.[108]

While price discounting by proprietary vendors relates to the anti-royalty provision of FOSS licensing, Microsoft's Shared Source Initiative might be seen as a response to the source code availability FOSS license provision. Although Shared Source is not open source licensing, it allows certain Microsoft customers to examine and study the source code of Windows and other products.[109] The

---

[106] *See* David Bank, *The Revolt of the Corporate Customer: How Companies Are Squeezing Tech Suppliers to Get a Bigger Bang for Their Software Bucks*, WALL ST. J., Jan. 17, 2005, at R1 (attributing the reduced price of proprietary software to, in part, the lower cost of open source software).

[107] *See* Free Software Foundation, Selling Free Software, http://www.fsf.org/licensing/essays/selling.html (last visited June 10, 2006) (touting the profitability of distributing free software).

[108] *See, e.g.*, Rebecca Buckman, *Microsoft's Malaysia Policy: As Poorer Nations Push PCs, Software King Lowers Prices in a Bid to Outfox Free Linux*, WALL ST. J., May 20, 2004, at B1 (discussing Microsoft's "rock-bottom prices" for computers running its software in Malaysia); Alisa Tang, *Microsoft Will Offer Low-Cost XP in Asia*, SPOKESMAN-REV. (Spokane, Wash.), Aug. 12, 2004, at 9A, *available at* 2004 WLNR 18303392 (discussing how Microsoft offered a stripped-down version of Windows XP at low cost to compete in the hopes of preventing users from switching to open source).

[109] *See* Microsoft Corp., Shared Source Initiative, http://www.microsoft.com/resources/sharedsource/default.mspx (last visited Aug. 16, 2006) (listing the initiative's various programs). The Shared Source Initiative program for "Enterprise" customers, for example, provides that "[l]icensees may read and reference the source code but may not

program's goals include enabling customers to better understand Windows, enhancing product feedback, and facilitating security, auditing, maintenance, performance tuning, deployment planning, and internal support.[110] These benefits are a subset of the benefits source code availability provides to FOSS development projects. The Shared Source Initiative explicitly acknowledges that its goal is to provide some of the benefits of open source software development, but within the paradigm of traditional commercial software development.[111] Two related forces triggered this response: the existence of FOSS alternatives mixed with some actual exit to those alternatives. In the Hirschman framework, the Shared Source Initiative (for those Microsoft products to which it applies) heightens customers' perceptions of the quality of the Microsoft products because the definition of what constitutes high quality software changed under the growing presence of FOSS alternatives and indirect voice about them.

More generally put, sometimes a product can remain the same, but external conditions will change customers' quality perceptions. This may characterize the deepening concern over computer security. Computers are increasingly interconnected through physical and wireless connections to the Internet. This creates a fertile environment for malware such as viruses and worms to hijack or disrupt computing resources from any class of users. Malware has primarily targeted a variety of Microsoft products because their ubiquity creates the greatest possibility of finding fertile hosts, i.e., computers with insufficient protections such as firewalls, antivirus, and other defensive capabilities.[112] In the case of the Microsoft Internet browser, Explorer, some reports suggest that customers are switching to a recently available FOSS alternative based on the perception that the FOSS alternative does not or will not suffer

---

modify it." Microsoft Corp., Enterprise Source Licensing Program, http://www.microsoft.com/resources/sharedsource/ licensing/enterprise.mspx (last visited July 15, 2006) [hereinafter Enterprise].

110 Enterprise, *supra* note 109.

111 Microsoft Corp., Basic Principles of Software Source Code Licensing, http://www.microsoft.com/resources/sharedsource/Articles/MicrosoftandOpenSource.mpx (last visited July 15, 2006).

112 *See* Andrew Beckerman-Rodau, *Ethical Risks from the Use of Technology*, 31 RUTGERS COMPUTER & TECH. L.J. 1, 17 (2004) (noting that Microsoft's operating system "is a favorite target of virus creators").

malware problems.[113] This disciplining effect did not arise from the FOSS alternative, but the urgency to make proprietary products more malware-resistant may have greater intensity due to the perception that FOSS alternatives do not suffer the same malady. A response is necessary to foreclose exit to the FOSS alternative, whereas without any alternative, in the Hirschman framework, this might be a "lapse in quality" against which a firm has insufficient incentive to quickly correct.

While there may be other disciplining effects from the exit or threat of exit to FOSS equivalents, these examples show that the dynamics identified in the Hirschman framework are active in the interplay between proprietary software and FOSS. Greater indemnification, price discounting, and Microsoft's Shared Source are examples of effects directly engendered by FOSS. Malware shows how FOSS equivalents, perceived to offer an equivalent set of functions and features, but without certain other problems, add to the disciplining effect proprietary firms feel in the face of the malware problem. These disciplining effects occur because exit is available. In the case of FOSS, the novel legal landscape has some potential to chill exit, but such chilling, to the extent it exists, seems at most to limit the acceleration of FOSS growth rather than forestalling it. Exit, as described above, is categorized among different types of users. These camps generate FOSS licensing approaches with differing mixes of exit and voice.

## II

## EXIT AND VOICE IN FOSS LICENSES AND PROJECTS

FOSS starts with a license. From the Hirschman perspective, the license is an institutional mechanism enabling exit and voice. It is

---

[113] Many users have switched to Firefox because they perceive it to offer better security. However, Firefox is not without security issues. *See Firefox Is Heading Towards Trouble*, EWEEK, Mar. 8, 2005, *available at* 2005 WLNR 3835018 (noting that Firefox is more secure than Internet Explorer, but still not "perfectly secure"); Antone Gonsalves, *Next Major Firefox Release Delayed*, TECHWEB NEWS, July 21, 2005, *available at* 2005 WLNR 11486883 (noting that the delay in Firefox's release was attributed to trouble "in the release of bug fixes"). By raising the data security issue, I do not mean to take a position as to whether FOSS or proprietary software is better in this area. This debate is ongoing, and is not my focus here, other than the role perceptions of data security play in a user's satisfaction with specific software applications. *See* Dennis Fisher, *Open Source: A False Sense of Security?*, EWEEK, Sept. 30, 2002, http://www.eweek.com/article2/0,1759,562220,00.asp (comparing market perceptions of data security between proprietary software and FOSS).

primarily based on copyright law but might also address patent rights. It typically grants, or seeks to clear, intellectual property rights to the extent possible to allow wide third-party latitude with the software.[114] This latitude includes the right to use, modify, and redistribute the code. The license facilitates software development resulting in code with a new, unique economic and social proposition: free use with available source code, under the condition that the FOSS requirements are observed and reapplied upon modification and redistribution.

The preceding Part of this Article provides an overview of the FOSS exit alternative to illustrate the dynamics a customer faces when considering the switch. This Part will fill in additional detail about this alternative from two related perspectives: (1) the license's mechanisms that allow exit, and (2) the development team characteristics a FOSS user can expect in comparison to traditional proprietary software development. The first perspective, the possibility of exit, also reflects voice. Sometimes this is indirect voice in the license, such as comments about the virtues of FOSS. More often, however, it is the direct voice of an exit threat that a viable FOSS alternative provides the end-user in dealing with proprietary software providers. The second perspective is an equally important component of the FOSS exit alternative. Users have greater structural mechanisms guaranteeing opportunities to participate in a FOSS software project as compared to traditional software development. The relationship between a software vendor and user is typically ongoing, and the degree of the user's investment in the technology determines each party's relative leverage.

### *A. License Rights and Language for Exit and Voice*

While all four major areas of intellectual property law might be used to protect software, FOSS licenses always address copyright law, sometimes address patent law, and occasionally address trademark law. Post-distribution trade secret protection is antithetical for FOSS software because the software is usually supplied with source code.

There are a variety of ways to categorize FOSS licenses, but here I will use these categories: corporate-style licenses granting

---

[114] *See generally* LAWRENCE ROSEN, OPEN SOURCE LICENSING: SOFTWARE FREEDOM AND INTELLECTUAL PROPERTY LAW 103-06, 126-28, 133-36 (2005) (discussing the way in which FOSS licensure developed and works).

copyright and patent permissions, the GPL and similar licenses, dual licenses, and attribution-only licenses. The analysis will not consider the last category because the license places very minor restrictions on use of the software and source code. Although some important FOSS projects operate under attribution-only licenses, these licenses merely claim copyright and then require that an attribution statement appear with the code.[115] The attribution-only license does not have the features to fully guarantee the institutional mechanism carrying exit and voice as I conceptualize these from the Hirschman framework. The subsections below address each of the other license categories.

### *1. Corporate-Style FOSS Licenses*

At the time of this writing the Open Source Initiative (OSI) listed about sixty licenses it deemed compliant with its certification criteria.[116] The Free Software Foundation's (FSF) list named about thirty other licenses not on the OSI list, although some were listed to show that they were not free software licenses.[117] These ninety licenses undoubtedly do not exhaust the list of licenses published or in use for FOSS.[118] Because this section does not need an exhaustive look at every license to make its points, I will draw examples from the OSI list. Among the sixty OSI-listed licenses, about twenty are attribution-only licenses, which I therefore put aside.

The majority of the forty remaining OSI licenses grant recipients rights under both copyright and patent law. Many are written in a style that clearly signals attorney involvement. An overall structural approach seems to have seeped into many of the licenses, perhaps

---

115 *See, e.g.*, Apache Software Foundation, The Apache Software License Version 1.1, http://www.apache.org/ licenses/LICENSE-1.1 (last visited July 15, 2006).

116 Open Source Initiative, The Approved Licenses, http://www.opensource.org/ licenses (last visited July 15, 2006).

117 Free Software Foundation, *supra* note 44.

118 *See* Ken Spencer Brown, *Open Source Serves Baskin-Robbins–Like Choices of Software: But It's Headache, Not a Treat; So Many Licenses Available, Companies Wrestling with 58 Flavors—and Counting*, INVESTOR'S BUS. DAILY, June 30, 2005, at A04, *available at* 2005 WLNR 10393503 (suggesting that the patchwork of licenses could threaten the industry's growth); Open Source Initiative, Charter for License Proliferation (LP) Committee of the Open Source Initiative (OSI), http://www.opensource.org/docs/policy/lpcharter.php (last visited Sept. 16, 2006) (explaining that OSI has created a committee "to identify and lessen or remove issues caused by license proliferation").

inspired by the licenses Netscape promulgated through the open source release and management of its browser code.[119] Two types of parties are typically defined: contributors and recipients. Any person or entity can be a contributor and recipient simultaneously. Recipients become contributors when they redistribute the software. Contributors grant a copyright license[120] and a patent license[121] to recipients. The licenses are conditional. They grant rights under the conditions that the recipient makes source code available and does not charge royalties upon distributing the software. Beyond this framework, however, additional conditions further define the character of the FOSS exit alternative presented by these licenses.

Many licenses demand compliance with patent and trademark terms. Names associated with the FOSS-licensed software, for example, might not be useable except under certain conditions.[122] Noncompliance with this provision might terminate the copyright and patent permissions granted by the contributor, or all contributors to the software for the noncompliant recipient. Both the copyright and patent license rights granted to the recipient may terminate if the recipient brings a patent infringement suit against other contributors

---

[119] *See* Mozilla.org, Mozilla Public License Version 1.1, http://www.mozilla.org/MPL/MPL-1.1.html (last visited July 15, 2006); Mozilla.org, Netscape Public License Version 1.1, http://www.mozilla.org/MPL/NPL-1.1.html (last visited July 15, 2006).

[120] *See, e.g.*, Eclipse, Eclipse Public License Version 1.0, ¶ 2(a), http://www.eclipse.org/legal/epl-v10.html (last visited July 15, 2006) [hereinafter EPLv1.0] ("Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form."). In some cases the non-patent grant of rights is stated more broadly as a license of intellectual property rights. Sun, Common Development and Distribution License (CDDL) Version 1.0, ¶ 2.1(a), http://www.sun.com/cddl/cddl.html (last visited July 15, 2006).

[121] EPLv1.0, *supra* note 120, ¶ 2(b). The granting language is as follows:

> Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents.

*Id.* The term "Licensed Patents" is defined as patents licensable by Contributor and infringed by the "Contribution alone or when combined with the Program." *Id.* ¶ 1.

[122] *See, e.g.*, Apple Computer, Inc., Apple Public Source License Version 2.0, ¶ 10, http://www.opensource.apple.com/apsl (last visited July 17, 2006) [hereinafter APSLv2.0] (discussing conditions for use of Apple's marks in association with the software).

or recipients. The reach of this "patent peace" clause varies among the OSI-listed licenses.[123] Some licenses present a broad reach. A patent suit by a recipient against a contributor or another recipient in any technology (including a technology wholly unrelated to the FOSS software) triggers termination of the plaintiff recipient's license rights.[124] Other licenses are less aggressive. They are content to terminate the plaintiff recipient's rights only when the patent suit is about the FOSS software or related technology.[125]

Although the corporate-style licenses typically grant both copyright and patent permissions, they also make clear that third-party rights may inhibit use of the software. The contributors disclaim indemnification and other guarantees that the software is infringement free, especially of patents.[126] Some licenses acknowledge that third parties may hold patent rights that inhibit use of the software.[127] These third parties may not use the software, and, as a result, there would not be a defense under the "patent peace" clause that the third parties have granted permission for the FOSS technology to infringe any claims in any patents held by such third parties. In other words, if a nonuser third party holds a patent covering the software, she can prohibit its use or require a royalty.

Some licenses explicitly allow the contributor or recipients to separately offer fee-based services such as support, updates, warranty, and indemnification. However, some of these licenses additionally require the service supplier to indemnify all other contributors for any service claims.[128]

---

123 *See, e.g.*, Sun, Common Development and Distribution License (CDDL) Description and Rationale, Executive Summary, http://www.sun.com/cddl/CDDL_why_details.html (last visited July 17, 2006) (discussing the narrowing of the "patent peace" clause in the CDDL compared to its predecessor license, such that the narrower clause covers only software released under the license).

124 *See* ROSEN, *supra* note 114, at 170.

125 *See id.* at 171.

126 *See, e.g.*, EPLv1.0, *supra* note 120, ¶ 2(c) ("As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.").

127 *See, e.g.*, *id.*

128 *See, e.g.*, APSLv2.0, *supra* note 122, ¶ 6 ("You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations and/or other rights consistent with the scope of the license granted herein . . . to one or more recipients of Covered Code. However, You may do so only on Your own behalf and as Your sole responsibility, and not on behalf of Apple or any Contributor.").

Thus, while FOSS is royalty-free and comes with source code, it carries a novel set of legal risks compared to traditional proprietary-licensed software. These risks explain the emergence of distributors, such as Red Hat, who provide the services discussed above. While Red Hat prices these services, their offering makes the FOSS exit alternative appear, from a licensing perspective, more equivalent to traditional licensing.[129] This business model of layering services on top of the FOSS license retains the benefits of source code availability and the price advantage of no royalties, while also normalizing the exit opportunity to make it more palatable to corporate information technology departments.[130]

As a result, many corporate users choose to purchase FOSS from distributors such as Red Hat, or procure systems from companies such as IBM that bundle FOSS into a set of goods and services. Some, but probably not all, of these users have the technical expertise and resources to download and install the freely available FOSS software. The comparatively greater expertise of the FOSS distributors to manage the distribution process, in conjunction with the layered services that they provide, channels users to the distributors. They optimize the FOSS exit alternative for corporate information technology departments. In this process, and in their presence in the marketplace, the FOSS distributors also contribute to the indirect voice in FOSS licenses and projects.

---

[129] Steve Ballmer, Microsoft CEO, relates: "As you point to the commercialization of Linux, which is going on, we are not competing typically [versus] 'free.' We are competing much more often with something else that has a positive price. So we are in a more normal competition." Carolyn A. April & T. C. Doyle, *It's a Microsoft World . . . Where Do You Fit In?*, VARBUS., June 24, 2005, at 28, *available at* 2005 WLNR 10608113 (quoting from a VARBusiness interview with Ballmer concerning the normalization of competition with FOSS).

[130] Open Source Risk Management touts itself as the "industry's only vendor-neutral provider of risk mitigation consulting and protection of open source users." *See A Legal Gun in the Open-Source Corral: With Users of This Software Vulnerable to Lawsuits, Venture Capitalist Daniel Egger Sees a Profit by Offering Protection*, BUS. WK. ONLINE, Nov. 12, 2004, *available at* 2004 WLNR 14506184 (discussing, in an interview with Open Source Risk Management (OSRM) founder, Daniel Egger, how OSRM hopes to help companies that use FOSS to get insurance against patent and copyright suits); Larry Greenmeier, *Service Offers On-Demand Tool for Finding Software-Licensing Violations: Black Duck Has Been Riding Wave of Concern Sparked by SCO Group's Lawsuits Tied to Its Claims on the Linux Code*, TECHWEB NEWS, Mar. 28, 2005, *available at* 2005 WLNR 4865139 (discussing how Black Duck Software, Inc. is offering software "designed to help companies identify open-source code being used in their IT environments and ensure that code is being used properly").

The corporate-style licenses are business oriented. In contrast to the GPL, discussed below, they have limited precatory language extolling the virtues or philosophy of FOSS. Even the few corporate-style licenses with precatory, voice-oriented language have only a modicum of such in comparison to the GPL. Traditional license agreements, like many contracts, sometimes have preambulary language describing the context of the transaction. Most corporate-style FOSS licenses omit this, although a few acknowledge the ideas behind FOSS before moving on to the substantive terms defining the rights and conditions. For example, one license by a large software company releasing a product as FOSS notes that it believes "that the open source development approach can take appropriate software programs to unprecedented levels of quality, growth, and innovation."[131] Another license states that FOSS "results in better quality, greater technical and product innovation in the market place and a more empowered and productive developer and end-user community."[132] Thus, a few licenses contain FOSS-advocating, indirect voice in the license text. Overall, however, the text serves the primary purpose to define the unique exit opportunity FOSS provides.

While voice-oriented language in corporate-style FOSS licenses is minimal, these licenses sometimes have parol materials extolling FOSS. These take the form of "frequently asked questions" (FAQ), lists, or similar writings posted on the web sites of organizations promulgating FOSS licenses. For example, a leading open source software package, called Eclipse, is licensed under the Eclipse Public License, which is a corporate-style FOSS license.[133] While the license itself does not recite the benefits of FOSS, the FAQs associated with the license discuss the business and technical advantages of the FOSS approach to software.[134] Ancillary materials such as licensing FAQs add to the indirect voice effect of

---

131 Computer Associates, Computer Associates Trusted Open Source License Version 1.1, License Background, http://www3.ca.com/Files/Licensing/trusted_open_source_license.pdf (last visited July 17, 2006).

132 Open Source Initiative, The Frameworx Open License Version 1.0, http://www.opensource.org/licenses/FW1.txt (last visited July 17, 2006).

133 EPLv1.0, *supra* note 120.

134 Eclipse, *supra* note 39, nos. 9 & 10. (First, "[a]n Open Source community provides a way for individuals and companies to collaborate on projects that would be difficult to achieve on their own." Furthermore, "[t]he Open Source model has the technical advantage of turning users into potential co-developers. With source code readily available, users will help you debug quickly and promote rapid code enhancements.").

the otherwise businesslike tenor of the corporate-style FOSS licenses.

As an institutional mechanism embodying exit and voice, these licenses favor exit. Indirect voice-content about FOSS philosophy is minimal. Like in the voice-laden GPL discussed below, direct voice is present in the threat of exit to a FOSS alternative; rather than complain, a proprietary software user tells the vendor she will switch if the vendor does not provide some feature or commercial benefit. Exit is viable only for those applications where FOSS equivalents are available. In these application categories, the mere possibility of exit adds to the voice effect arising from the license and software.

Many of the corporate-style licenses originate from companies that fit, at least to a degree, in the classification of "open source advocates." In several instances, these companies "donated" entire software products or technologies to the FOSS movement and made the source codes available under a corporate-style FOSS license.[135] These instances endow FOSS users with additional exit alternatives, enrich the overall FOSS code base, and help increase the greater FOSS community. Even non-FOSS users are likely intrigued when suddenly a new FOSS alternative appears. The intrigue may derive from the possibility of lower cost, some identity with the message of FOSS, hopes for more influence over a software technology, or a combination of all of these benefits.

### *2. The GPL*

Compared to the corporate-style FOSS licenses, the tenor of the GPL and its related licenses is counter-establishment. More than any other FOSS license, the GPL leads with activism. At the same time, it originated the unique FOSS exit alternative. It has spawned some related licenses, but the important points can be made simply by working with version two of the GPL, although much of this discussion also applies to the draft of version three posted in January 2006. The GPL's voice-filled preamble is almost a full page of the GPL's seven pages.[136]

The preamble opens with the proposition that traditional licenses "are designed to take away your freedom to share and change . . . software" and then goes on to explain, for several paragraphs, how

---

[135] *See* Hamerly & Paquin, *supra* note 15, at 203-06 (describing Netscape's release of its code to the public at large).

[136] GPL, *supra* note 7, at Preamble.

and why its copyright-based licensing system corrects these traditional inequities.[137] Moreover, the preamble also discusses the threat to FOSS exit it contemplates from software patents.[138]

The substantive terms of the GPL attempt a "plain-English" approach to legal drafting. Lawyerly drafting practices such as defined terms and other mechanisms are minimally used. Most of the GPL sets out the copyright-based license conditions of source code availability, no royalties, and reapplication of the same terms upon distribution of the same or modified versions of the software.[139]

Version two's substantive language for software patents, however, does not explicitly implement the preamble's indirect voice content about patents. In essence, term seven of version two of the GPL only says that a recipient cannot distribute the software if doing so would contravene some other legal prohibition, such as in the case of patent infringement.[140] From the software patent perspective, the corporate-style FOSS licenses provide an exit alternative with greater minimization of the exit-inhibiting risks that might arise from third-party software patents. Here, however, it is important to distinguish version two of the GPL from version three because the latter version implements explicit patent permissions.[141] Thus, the final draft of GPL version three may provide license equivalence in this area.[142]

---

137 *Id.* ("To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.").

138 *Id.* ("[A]ny free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.").

139 *See* ROSEN, *supra* note 114, at 105-07, 125-33 (explaining, in detail, the structure of the GPL).

140 GPL, *supra* note 7, § 7. License term eight has a similar provision for a special case when distribution in a particular geography is not available due to blocking patent rights. *Id.* § 8.

141 GPLv3, *supra* note 7, §§ 2, 5, 11.

142 In the case of either the GPL or corporate-style FOSS licenses, proprietary software vendors represent the most likely group of potential patent infringement plaintiffs. Additionally, merely obtaining a patent license to use, modify, or distribute a program would not lead one to characterize the program as "proprietary," unless perhaps the license was exclusive.

Even though unorthodox, the success and popularity of the GPL is undeniable. It was the first FOSS license,[143] and it has a strong message of indirect voice. The FSF refers to the GPL's indirect voice content as the "constitution" for the free software movement.[144] It is probably the most widely used FOSS license at the time of this writing.

From the perspective of Hirschman's *Exit, Voice, and Loyalty*, the GPL's success derives from a synergistic concentration of exit and voice. The Hirschman framework suggests that sometimes society-wide benefits result when institutional mechanisms facilitate greater voice or allow exit under protest.[145] The GPL seems to be a unique example of an institutional device using both exit and voice to discipline an entire industry. Hirschman's framework is about both economics and politics because he is interested in the disciplining effect of exit and voice, independently or in conjunction when both are present, on both firms and non-firm organizations, including governments.

The FOSS movement is both political and economic, especially considering the emphasis of the two "camps." While free software advocates are most interested in the social and political advantages wrought by FOSS licensing, the open source software advocates emphasize economic integration of FOSS into the greater information technology infrastructure. The synergistic dualism that starts with the GPL, and maps nicely to Hirschman's framework, is reflected in the FOSS community. The political message of the GPL goes beyond competitive factors of software functionality, even as its licensing terms define an exit alternative that may reorder large swaths of a critical industry. The GPL's creed of functional freedom, through software sharing, invites an evaluation of its social value for software technology.[146]

---

143 *See* MOODY, *supra* note 60, at 26-28 (describing Stallman's creations as the "main engines in driving the free software projects on to their extraordinary success"); ROSEN, *supra* note 114, at 103 (describing how the GPL transformed the world of software).

144 MOODY, *supra* note 60, at 27; Dixon, *supra* note 76, at 106 n.257; Tai, *supra* note 76.

145 *See* HIRSCHMAN, *supra* note 1, at 119.

146 *See* Jonathan Zittrain, *Normative Principles for Evaluating Free and Proprietary Software*, 71 U. CHI. L. REV. 265, 268, 274-75 (2004) (discussing FOSS as a social movement: "the legal system must have a framework with which to judge the social value of free software's open development model").

The GPL's indirect voice content is much greater than that of the corporate-style FOSS licenses. Both license types provide exit for users who need software for their operations. The traditional FOSS bargain, however, does not allow those users to privatize the software through traditional royalty-based licensing. Only in the rare case of permission from all contributors could one privatize the code. In such a scenario, any current user still would probably be able to take the code down an open source path. Given this situation, some companies created a new type of licensing system, called dual licensing, that builds on the ideas of FOSS licensing and allows distributors a choice as to whether or not the code they distribute is open source.

### *3. Dual Licensing*

Dual licensing works as follows: if a distributor uses a FOSS license with her users, then the originating dual licensor provides the software under a FOSS license. On the other hand, if the distributor takes a non-FOSS approach, licensing only object code and charging royalties, the dual licensor applies traditional, royalty-bearing, proprietary software licensing terms.[147] In essence, the dual licensor offers bifurcated terms, and the distributor-licensee chooses to operate on one side of the bifurcation or the other. The originating dual licensor, however, can incorporate software revisions it finds on the open source side into the proprietary side.

As a hybrid license that charts a path between a FOSS license and a traditional software license, dual licenses retain some of the attributes of the traditional proprietary licensing scheme. This arguably enables companies whose business models are not based on FOSS complements, such as Red Hat and IBM, to prosper with a licensing revenue stream.[148]

From the perspective of the dual licensor, the other benefit is that it can in effect "harvest" code from the open source community and include the harvested code in the original software project for future licensing under either a FOSS model or propriety terms. The originator's permission to do this is in the original dual license. Under this structure, as soon as a FOSS licensee of the dual-licensed

---

[147] *See, e.g.*, MySQL, MySQL Licensing Policy Version 5.1 http://www.mysql.com/company/legal/licensing/index.html (last visited July 17, 2006).

[148] *See, e.g.*, Mike Olson, *Show Me the License*, LINUX WORLD MAG., Aug. 11, 2003, http://linux.sys-con.com/read/33893.htm.

software distributes the code, the FOSS side of the dual license requires source code availability, and the dual license also allows the originator to incorporate the code into the master software project. While not every modification to the original code will be distributed, thus triggering source code availability, the structural benefit of the dual license is that the partial commons created by a FOSS license is available to the originator for relicensing under commercial terms on the other side of the dual license, so long as the originator also makes the code available under the FOSS license.[149]

The dual license provides an exit opportunity that might be FOSS, or might not. It is significant for the exit choice it presents software integrators, distributors, and value-added resellers (VARs), but dual licenses are less well-suited for end users. The details of dual licensing are complicated and they have greater applicability for certain types of software, such as code designed as a component for other software.

As a unique innovation of the FOSS license, the dual license expands the FOSS-like exit opportunities originating from the movement. To the extent the dual licensor promotes and supports a FOSS development community around the software technology, end users have greater possibilities for viable FOSS equivalents for exit. The dual license acts as a cross-subsidization mechanism whereby license-paying distributors support the dual licensor's business, allowing the company to promote the FOSS community as well as develop the product itself. These distributors may have customers to whom they can apply the traditional royalty-bearing licensing model because the customers lack the greater technical expertise sometimes necessary for FOSS, or because the distributors have value-added technology that they bundle with the dual-licensed software.

Besides expanding FOSS exit opportunities, dual licensing carries indirect voice about the merits of FOSS. For example, one well-known dual licensor declares, in its FOSS license, that the "intent of this license is to establish freedom to share and change the software

---

[149] To complete the "quid pro quo" sometimes used to justify dual licensing, and perhaps mollify FOSS purists who dislike dual licensing, the originating dual licensor who incorporates distributed FOSS modifications into the master code base must continue to make the third-party revisions available under both sides of the dual license. *See* Robert W. Gomulkiewicz, *Entrepreneurial Open Source Hackers: MySQL and Its Dual Licensing*, 9 COMP. L. REV. & TECH J. 203, 209-11 (2004) (describing dual licensing generally, and describing specifically MySQL's dual licensing implementation, which included a need to handle license compatibility issues arising from the GPL).

regulated by this license under the open source model."[150] This dual licensor notes the following in its licensing overview materials:

> Trolltech aims to make the best cross-platform development tools in the world. By selling commercial licenses, we are able to staff a full-time dedicated development team and are able to provide first class support.
>
> By providing our products under open source licenses, we are also an active member of the open source community. This community has played an important role in ensuring the stability and quality of our products. Trolltech's products are thoroughly tested by thousands of open source developers around the world. Through active community participation in our development process, Trolltech products reach commercial stability far more quickly. We call this our Virtuous Cycle.[151]

The indirect-voice content in dual licenses and their supporting materials tends to match the volume and tone of the corporate-style FOSS licenses, and thus pales in comparison to the indirect-voice content of the GPL. The dual license joins the other two categories, however, in offering an exit alternative with software development transparency and a perception that users will have greater voice in the progression of the software.

### *B. FOSS Development Transparency*

FOSS software development depends on a variety of group organizational practices that are not necessarily encoded in the FOSS license.[152] These practices differ from traditional proprietary software development but must accomplish the same objective: allowing groups of programmers to work together to generate interoperable software that comprises a software product or technology.

#### *1. FOSS Project Governance and User Participation*

It is becoming increasingly difficult to stereotype FOSS development, but for the most prominent projects there are some

---

150 Trolltech, Q Public License Version 1.0, http://www.trolltech.com/products/qt/licenses/licensing/qpl (last visited Sept. 17, 2006).

151 Trolltech, Business Model, http://www.trolltech.com/company/about/businessmodel (last visited July 17, 2006).

152 See WEBER, *supra* note 37, at 72-82, for a good discussion of the eight "general principles that capture the essence of what people do in the open source process."

common elements.[153] One oft-celebrated feature is its distributed nature. The programmers are scattered, possibly across the globe, and use the Internet to coordinate activities. Volunteerism, or at least subsidization, fuels the projects. Either or both could come from an individual or company. FOSS's volunteer-based, distributed-development model is also unique in its opportunity for programmers to self-select for work within the project.[154] While programming is as much art as science, good solutions are recognizable. One can earn one's way onto a desired part of the project by contributing superior code for that part.

Each project has some measure of leadership, comprised of one or more individuals, but typically not a large group. The leadership group comes together in a variety of ways.[155] Among these leaders, various forms of group decision making might apply, such as the formal backdrop of corporate governance if the project is housed within a nonprofit, codified bylaws or other governance procedures, or informal persuasion and deference among the group leaders.

Deliberations, along with just about everything else relating to the project, tend to be publicly available via web site(s) devoted to the project.[156] With sufficient technical acumen, users, programmers, and the general public can examine bug fix submittals from users and developers, discussions about new functionality and the timing of the release of new versions, and other internal matters. This transparency into the inner workings of the development process is a key distinguishing factor compared to traditional software development. It is an important factor in the exit and voice a FOSS-equivalent technology provides.

The incentive structure of the development team, and its host organization, impacts software users. Proprietary software users may see advantages in FOSS development teams. Not only can the FOSS user see the code as it develops, often she can review the

---

153 Many, but perhaps not all, of these common elements of FOSS development will apply even when the project springs from a dual licensor or FOSS distributor.

154 *See* Yochai Benkler, *Coase's Penguin, or, Linux and the Nature of the Firm*, 112 YALE L.J. 369, 414-15 (2002) (noting that an advantage of open source software and peer production is that, compared to management hierarchies, contributors are better able to judge where best to apply their talents within various projects).

155 See WEBER, *supra* note 37, at 88-93, 166-71, for an overview of the various leaders and styles of leadership within the FOSS community.

156 *See, e.g.*, The Linux Kernel Archives, http://www.kernel.org (last visited July 17, 2006).

deliberations of the development team, communicate with the developers directly, and make suggestions for bug fixes, features, and functionality at a much greater level of detail and technical sophistication due to the source code availability. The user's direct voice, after switching to FOSS, may be more potent because the traditional corporate intermediary no longer separates the user from the development team.

Moreover, the FOSS-switching user knows that if necessary, she can take the development in-house if the developer community disbands or loses interest. This is not necessarily a panacea: most users want their software product providers to remain viable to provide, at the least, upgraded software versions. On the other hand, taking over a FOSS-community supported project is likely a less difficult exercise than the equivalent doomsday scenario when a proprietary software company or product dies, triggering a release of source code under the escrow agreement.

### *2. Project Abandonment and the Insufficiency of Source Code Escrow*

Built into FOSS licensing is a better remedy for the doomsday scenario of development team abandonment than the traditional source code escrow approach. Even though its benefits are hard to realize, a market exists to place software source code in escrow. This happens mostly among corporate entities for high-value or negotiated software licenses. In order to make the sale, many proprietary software providers must represent to corporate users that the source code is available from escrow. The provider enters into an escrow agreement with a third party. If the provider abandons the product, as defined in the escrow agreement and typically meaning that the provider no longer exists as a going concern, the corporate user has the right to obtain the source code for purposes of internal maintenance and support. Traditionally, the corporate user did not have broad latitude with the source code once released from escrow.

The other problem with source code escrow was that it relied on the continued diligence of the software product provider. As the object code of new releases goes to users, the source code for the release should go to the escrow company. But it might not,

especially if the software company is under financial pressure.[157] Even if delivered, the quality of code commenting (sometimes essential for a third party to deal with the code) may drop during times of financial decline. In sum, there are a number of reasons why an abandoned user might be disappointed and without an effective remedy when she turns to the escrowed copy of the source code.

Contrast source code escrow with FOSS, where a current copy of the source code is always available and can easily be obtained by the user. If a corporate user will not need to resell the software because it is merely an operational resource and not a profit opportunity, FOSS is superior to combat the doomsday scenario against which source code escrow agreements are meant to protect. The user can monitor code quality in an ongoing manner. It does not have to wait until it is too late to discover that the code is so poorly commented, written, or designed that maintenance costs will be outrageous, if the software can be maintained at all. Moreover, the user is more likely to be able to discover the identity of programmers the user may want to hire to continue development and maintenance. In hiring the programmers, the users have the option to simply maintain its own internal version, attempt to revive the FOSS community around the software, or create a new community. The dissatisfying nature of source code escrow enlightens the advantages FOSS provides to solve the abandonment problem in a way that maximizes the user's chances to retain a vibrant and viable code base.

Abandonment is a possibility in both proprietary software and in FOSS. The user's incentives arising from possible abandonment of FOSS, however, help counter the non-contributing-user problem inherent in FOSS and thus coloring its exit opportunity. A user can download FOSS and use it internally without any obligations to pay anyone or contribute anything to the project. A high percentage of

---

[157] *See* Jon C. Christiansen, *Doing Software Escrows Right*, 21 COMPUTER & INTERNET LAW. 17, 17-18 (2004), *available at* http://www.escrotech.com/Doing Software Escrows Right - 2.pdf (discussing licensees' lack of knowledge on how to use the source code, third-party ownership of the software, and improperly maintained or updated escrows as some of the common problems of source code escrow); Dean Gloster, *Typical Source Code Escrow Agreements: What's Broken and What Works Instead*, FARELLA BRAUN & MARTEL L.L.P., May 25, 2005, http://www.fbm.com/index.cfm/fuseaction/publications.archive/publications_archive.cfm (filter publications by "Bankruptcy and Creditors' Rights"; then follow "Typical Source Code Escrow Agreements" hyperlink) (discussing the practical reasons why source code escrows do not work).

such users in the population for an application can diminish the import of some of the most important benefits of FOSS development practices, which inherently work better with a more active user base. For example, one explanation to support the claim of FOSS's higher quality is that a "massive" peer review process helps vet the code in a way not available in traditional development.[158] This process is less effective as noncontributing users increase in the user base.

A FOSS user, however, knows that in many cases the development team has a more fragile persistency compared to traditional software development groups. This is especially true for FOSS projects without an organizational anchor, such as a nonprofit, dual licensor, or corporation with complementary services. This knowledge drives an incentive to participate in the community by providing input to the development team, helping write software manuals, submitting bugs or even suggesting code revisions to fix problems if the user has the technical acumen, and generally remaining in touch with the development community.

There is a similar incentive to preventively participate in such activities with proprietary software as well, but the opportunity for participatory scope is reduced, and the perceived need is likely different. Many corporate users pay regular maintenance fees to their software providers. With these payments, it is easy for a user to take the attitude that all she needs to do is pay the fees and report problems when she perceives them. This arrangement deemphasizes the opportunity for the corporate user's personnel to contribute as frequently in a deep and meaningful way. While these personnel are sometimes able to help advance the software if they had access to internal information, such access is often limited with proprietary software. FOSS licensing, on the other hand, makes all internal software information available. The curiosity of a user's personnel, combined with the knowledge that involvement helps sustain the community that brought forth the code in the first place, invite involvement. This involvement helps prevent the demise of the FOSS development community.

### *3. Responses to Disbanding Development Teams*

One path to demise for a FOSS community is that it loses energy and disbands, either because the software attracts insufficient

---

[158] *See* Raymond, *supra* note 63, § 7.1 (discussing the positive effects of "massively parallel peer review" for software development).

numbers of participatory users, or because the leaders lose interest and no new leaders emerge. Another path to a FOSS community's demise is the often discussed but rarely occurring "fork" where some members of a development team exercise their rights under FOSS licensing to exit the original team and chart a new path with the software.[159]

The fork possibility illustrates that Hirschman's exit and voice mechanisms influence dynamics within a FOSS development group[160] as well as between the proprietary software user and her vendor. While the processes of exit and voice within a FOSS community are not this Article's focus, I touch upon them briefly because they color the exit opportunity for the user switching to a FOSS equivalent. Just as the FOSS license enables both exit and voice for the user against proprietary software vendors, it enables exit for any developer or group of developers within a FOSS development community. There are factors that limit the occurrence of a forking exit, but in theory, it is possible.

FOSS licenses inherently allow a project to fork; that is, one group of developers can take the code base and start down a different path. This group would exit the development collective of the original in order to strike this new path. The common wisdom is that forking is rare, but the structural point, in light of the Hirschman framework, is that forking is a possibility of exit with a disciplining effect on the development project leaders.[161] There is no equivalent disciplining effect on the development teams for proprietary software; disgruntled programmers can change employers or try to change assignments, but the developers are without the ready legal rights to "fork" the project as compared to the FOSS developer.

A number of factors produce an incentive structure that helps limit forking. First, the governance of most FOSS projects establishes a norm of transparently debating and working out problems.[162] Even if programmers are not directly involved in an issue, they can always review how it was "adjudicated" in the communications typically logged on the Internet in relation to the project. In that sense, the character of the leaders is always open to public inspection, which

---

[159] *See* ROSEN, *supra* note 114, at 301-03 (describing "forking" generally, and suggesting, as an example, Sun's SISSL as a model to prevent it from occurring).

[160] *See* WEBER, *supra* note 37, at 158-60.

[161] *See id.*

[162] See Coleman, *supra* note 49, at 6-8, 24-28 for an example of how debate functioned within the Debian project.

can give group members confidence that some direct voice is worth the effort.[163] Second, project leaders often need to recruit developers and users. Their recruiting investment in the group makes them more likely to compromise before losing a part of the development team to a fork. Third, momentum is an obvious force that might limit forking, and Hirschman notes that it is a general inhibitor of exit.[164] Fourth, developers understand that the sum is greater than the parts. A forking group who only takes part of the team may not achieve its objectives. It may have insufficient resources to chart the new path it desires.

As predicted in the Hirschman framework, when exit is limited, voice often plays a greater role in disciplining an organization. It is likely that both mechanisms influence FOSS development teams. Exit is not optimal, and direct voice is easy to implement. The community's institutional structure is uniquely built to easily take inbound communications. Developers use this structure to create the code, and can use it to express dissatisfaction with the organization. An active FOSS development team is thus accustomed to processing and responding to direct voice. This capability is an attractive feature for proprietary users considering a switch to a FOSS alternative.

Switching to FOSS is attractive due to perceived advantages of transparency for the development process, even if the persistency of the programming team is potentially fragile in comparison to proprietary software vendors. As the FOSS ideology gains greater overall acceptance, the fragility of any particular FOSS project may lessen. Beyond the chance to participate to a greater degree in development, FOSS exit offers the proprietary software user the other mainline advantages of the FOSS license: royalty-free use, source code availability, and conditions that try to ensure the continued survival of the first two terms. Led by the GPL and its related licenses, a variety of FOSS license types implement this basic FOSS promise. The license categories differ in their approach to technical issues, such as whether they forestall assertion of patent rights by recipients of a FOSS technology. All FOSS licenses claim copyright in the code for the contributing developers or their

---

163 *See id.* at 58, 62-69 (chronicling the monumental response to one seemingly benign e-mail and how it nearly tore the Debian project apart).

164 *See* HIRSCHMAN, *supra* note 1, at 78 (discussing how loyalty to an entity dissuades exit).

assignees,[165] and the categories discussed in this Part provide a FOSS exit for users of proprietary software. Each does so with varying degrees of indirect voice springing from either the license text or from materials associated with the license. The voice-carrying capacity of the FOSS license, and in particular the path-breaking GPL, highlights its unique character as an institutional mechanism that symbiotically combines exit and voice, offering a new response to perceptions of decline in traditional proprietary software.

Parts I and II of this Article focus on the proprietary software user considering the switch to a FOSS alternative. The dynamics of this exit opportunity have: (1) elements of direct voice in the actions or communications by the user directed to her proprietary vendor, and (2) elements of indirect voice, such as advocacy to wider audiences that a change is needed.

The next two Parts expand on the indirect-voice theme within the Hirschman model, although some related examples of direct voice are also briefly discussed. Each of the next two Parts intrinsically depends on the existence of the exit mechanism as well. The presence of viable FOSS alternatives makes threat of exit a potent message that FOSS advocates can deploy as direct or indirect voice.

## III
## EXIT, VOICE, LOYALTY, AND NEGLECT—THE EXTRACURRICULAR FOSS CONTRIBUTOR

Among the areas where Hirschman's framework has influenced legal scholarship is labor and employment. In that context, the employee has the options of exit or voice, and, if conditions allow for it, loyalty may forestall exit to allow voice to operate.[166] Scholars have extended the Hirschman framework to add neglect, where, in the employment context, the employee does not exit, but declines to voice dissatisfaction.[167] The first section of this Part briefly reviews this extension and its applicability to programmers and information technology personnel. The second section applies the extended framework to the moonlighting FOSS contributor.

---

165 ROSEN, *supra* note 114, at 28-30.

166 *See* HIRSCHMAN, *supra* note 1, at 77-81 (discussing the role of loyalty in his framework generally).

167 *See, e.g.*, Rusbult, *supra* note 52, at 601-02.

### *A. Neglect as an Extension to the Hirschman Framework*

The employment context illustrates subtleties in the original *Exit, Voice, and Loyalty* framework. Exit and voice each have an opposite. Employees can stay rather than exit, and remain silent rather than give voice.[168] Hirschman posited loyalty as a mechanism that forestalled exit, allowing voice to operate in some situations.[169] Specifying the loyalty mechanism can be difficult because the circumstances compelling an employee to stay and give voice might not fit into general conceptions of loyalty, perhaps reducing that word to a label for anyone who stays and voices for any reason.[170] Hirschman's exposition did not emphasize the combination of no-exit (stay) and no-voice (silence). This is neglect, where the dissatisfied customer, member, or employee stays and suffers her dissatisfaction in silence.

A technical employee might be dissatisfied with her job for any of the usual reasons unrelated to her areas of expertise, but I want to focus on dissatisfactions programmers and other IT professionals may feel that relate to their technical opportunities and sense of professional identity and community. These may be attractions for FOSS contributors, and to the extent they are lacking in their job, there is the possibility of finding them in extracurricular FOSS work. On the other hand, FOSS may not provide such hypothetical salve. This Part presents these possibilities as a framework to conceptualize exit and voice for the extracurricular FOSS contributor, recognizing that empirical work is necessary to verify the framework or its intuitions.

Within computing, there is a tremendous amount of legacy code. This is old software written in outdated programming languages. Programmers maintaining these applications may have little chance to learn new languages and software technologies on the job. The company's need to retain a productive specialist in her current role may diminish the company's incentive to move technologists to career-enhancing positions. These spots may be filled by employees with more recent training and familiarity with newer technologies. The resulting technological entrapment the programmer feels may be bearable given the other benefits of employment, yet nonetheless the

---

168 *See* Laver, *supra* note 31, at 471, 477-81.

169 HIRSCHMAN, *supra* note 1, at 77-78.

170 *See* Laver, *supra* note 31, at 480-81 (questioning the extent of the relationship between voice and loyalty).

situation remains dissatisfying. The offsetting benefits might include a community of peers, or it might not. This could depend on whether the company generally employs other programmers, or on whether its software applications require team-intensive development.

The job satisfaction of every programmer will not necessarily rise and fall on career enhancing and peer community opportunities, but the unique nature of programming helps explain why these are often important and desired satisfactions. The work is a unique deployment of human capital. It is complex, creative, and often team-implemented. Given the nature of software development, a programmer's mindset, enthusiasm, commitment, and energy for the project is critical for her employer and serves as a barometer of her satisfaction. The extracurricular FOSS work may signal that the technologist has withdrawn to a diminished state of commitment for her employer's project. This posited lost energy is indirect exit. A potential cause is that the FOSS work can sometimes ameliorate these dissatisfactions.

FOSS development often uses current technology. One can work with, or even help create, some of the newest software technologies, breaking a cycle of technological entrapment. Also, FOSS offers a strong tradition of community and peer involvement. Even though many FOSS communities are virtual, they provide the FOSS contributor with an affiliation that often presents philosophical attractions. While there are a variety of other motivations for FOSS developers, these two are possible salve for the dissatisfied technologist who cannot exit her full-time job. This balm does not reverse the employee's state of neglect, but it provides an extracurricular outlet with countervailing benefits.

### *B. Voice from Extracurricular FOSS Contributions*

Conceptually, the moonlighting FOSS contributor might fit within the neglect category of the extended Hirschman framework. From the perspective of the relationship between employer and employee, by definition, a programmer in the state of neglect is not exercising direct voice. Even if this is true, however, the FOSS contributor's moonlighting has indirect voice value in the dispositions generally occurring between software users and proprietary software vendors, and possibly within the contributor's employer. Conceptualized in the extended framework, the dysfunctional state of the employer-employee relationship is a dynamic factor in the general ethos of

opinion about closed versus open software. In other words, the neglect may spill over as indirect voice with a general, and perhaps specific, effect on the contributor's employer.

The proposition that moonlighting on FOSS generates indirect voice within the software ecosystem relies on three points. First, there are a nontrivial number of gainfully employed technologists who voluntarily contribute to FOSS apart from their primary job.[171] Second, some of these technologists work for traditional proprietary software providers, although the technologists who work for end users also fit into the equation. Third, there is some degree of implicit or explicit disclosure about the moonlighting to others, including perhaps to the technologist's employer. The mere fact of the moonlighting, and others knowing about it, is the primary fount of the indirect voice.

The argument for this proposition does not assume an extraordinary programmer doing spectacular things in the code or within the FOSS community. Instead, an ordinary FOSS volunteer is the focus, someone who puts in a few hours a day, week, or month.[172] Nor does the contribution need to be code. I include in the definition of the FOSS contributor nonprogrammer information technologists such as system administrators, quality control and testing personnel, and related roles. The key distinction in this framework is that their extracurricular activity contributes to the FOSS project and rises above the activity of a passive user. For example, a nonprogrammer who applies the Linux kernel to novel hardware, regularly discovers problems, and submits bug reports to the relevant hierarchy contributes without programming.

Under the first point, common lore for FOSS states that a nontrivial, and perhaps substantial, amount of the programming labor on many FOSS projects is a volunteer effort.[173] At this level of generality, the assertion is hard to dispute, although it is difficult to quantify empirically. The analysis should exclude technologists who are paid to work on FOSS by complementary providers such as Red Hat or IBM. More generally, the analysis should probably exclude technologists whose primary means of financial support is

---

171 *See* MOODY, *supra* note 60, at 154-55; WEBER, *supra* note 37, at 130-33.

172 It is not the purpose of this Article to set a time commitment threshold that would define an "ordinary extracurricular FOSS contributor." Rather, this Article assumes that such a commitment would be secondary to the technologist's employment.

173 *See* MOODY, *supra* note 60, at 154-55 (suggesting that the motivation of hackers is similar to that of famous artists).

highly complementary to FOSS. Many other viable motivations serve to explain why there would be a population of FOSS moonlighting contributors, including the motivation to develop one's skills with new or different technologies, and the motivation to scratch an itch—that is, create some feature or function for one's own use.

The second point is that FOSS moonlighting contributors are employed both by end users, and paradoxically, proprietary software vendors. There is a degree of indirect voice in each case, although the second case has greater shock effect.

Consider the case where the moonlighting technologist is employed by an end user. This situation does not pose the direct conflict of philosophies inherent where a proprietary software vendor employs the technologist. On the other hand, moonlighting by any employee often creates risks for the employer. As a result, some end user employers seek to prohibit or discourage moonlighting. One legal risk for end-user technology companies arises from the potential disclosure of proprietary technology by the moonlighting activity. Another disadvantage from a human resource perspective is that, at some level, moonlighting diverts energy from the technologist's primary employer. Some employers flatly prohibit moonlighting, but of course, not all employees comply.[174] Other employers allow it under various conditions, which range from requiring authorization, to quantifying the maximum number of moonlighting hours or specifying the technologies within which the employee can moonlight.

Under the third point establishing how FOSS moonlighting might generate FOSS voice, disclosure of the employee's moonlighting may occur formally if her employment agreement or policies require employer authorization or notice to the employer. Informal disclosure both to management and to coworkers is also possible within the social circles inherent in most workplaces. If the technologist discloses her FOSS moonlighting, this can function as indirect voice due to FOSS's aura. The indirect voice effect would be stronger if the employee is vocal about her reasons for

---

[174] An employer's moonlighting prohibition may be explicit, or implicit due to the employment contract typically vesting intellectual property ownership with the company. Thus, all ideas in a field of technology, or ideas related to the company's operations, will belong to the company. This can preclude the legal right to contribute to FOSS without a release from the company if the contributed code embodies intellectual property belonging to the company.

contributing or perceptions of FOSS's benefits or values. Many professional workplaces have vibrant social interactions well beyond business concerns. This workplace social structure is a plausible place to learn about and debate the pros and cons of FOSS.

The analysis thus far assumes that the moonlighting FOSS technologist is in the state of neglect, where she does not voice her job dissatisfaction. Removing this assumption, the technologist might apply direct voice when employed by corporate end users, meaning that she may advocate to her employer to begin or increase the use of FOSS.[175] If moonlighting contributions to FOSS help sooth her dissatisfaction, it is a logical path for her to internally advocate greater use of FOSS. This direct voice, if heeded, would help her recuperate greater satisfaction from her job, assuming she is involved with the FOSS implementation at work. The end-user gets the benefit of a fully energized employee, along with the various other benefits of the FOSS value equation that apply to its situation.

This same analysis may not be possible if the employee works for a proprietary software vendor, especially one with viable FOSS competition. The natural inclination of the management of such an employer would be to frown on extracurricular FOSS contributors. Management unhappiness with FOSS moonlighting might be less severe if the company's products have no FOSS competitors and the FOSS contributor works on complementary technology. For example, suppose the moonlighting contributor works on the Linux kernel, while the company's software product is for engineering design for aerodynamic air flow systems. The company might very well sell more copies of its software, assuming it is available on the GNU/Linux operating system, due to the existence of that operating system if the combination provides a lower-cost platform.

It also might be the case that the dissatisfied and moonlighting FOSS technologist would exercise some direct voice to management to adopt FOSS for internal operations. If the company's proprietary software product is the aerodynamic design software, what is the harm, the employee might ask, to using GNU/Linux to run the company network, e-mail server, web server, and firewall, and reduce operational costs by doing so? It seems unlikely, however,

---

[175] *See, e.g.*, C.J. Kelly, *Eyeing an Opening for Open-Source: Our Security Manager Is Surprised When Her Boss Takes an Interest in Exploring Some Open-Source Security Options*, COMPUTERWORLD, July 4, 2005, at 25, *available at* 2005 WLNR 11479060 (describing how a computer security technologist influenced supervisors at the company to deploy FOSS for various internal network infrastructure projects).

that the employee would expend direct voice to convince the company to convert its revenue-source software product to FOSS. That does not mean, however, that the social knowledge of the employee's FOSS moonlighting does not have indirect voice effects.[176]

Within the Hirschman framework, several structural points arise from these scenarios. First, by taking a path of indirect exit, i.e., operating in a state of neglect, a dissatisfied moonlighting FOSS contributor generates indirect voice about FOSS. This indirect voice may carry forth as positive advocacy and radiate within the professional and social circles that the contributor inhabits. Anyone who knows the technologist and her activities, including her coworkers, will learn about FOSS.

Second, for corporate users and proprietary software employers, the dissatisfied employee may try direct voice and attempt to convince her employer to adopt FOSS to some degree. Unlike most hobbies, FOSS has features that could be attractive to proprietary software employers. It does not necessarily matter that the moonlighting FOSS contributor works on a project that most companies would not use in their operations, such as gaming software. The principles of FOSS licensing apply across differing applications. It is the understanding of those licensing principles that the moonlighting employee could convey.[177]

The moonlighting FOSS contributor illustrates the Hirschman framework in the employment setting. Employment relations and employment law are areas where the Hirschman framework has appeared in the literature.[178] FOSS applies to the employment context by offering the employee in a state of neglect a two-prong outlet. First, she feeds her creative needs by moonlighting on FOSS

---

176 *See Community Debates Microsoft's Open-Source Agenda*, EWEEK, June 3, 2005, *available at* 2005 WLNR 9961029 (reporting a software industry analyst as stating that "there are substantial bodies of people within Microsoft that either already have or are ready to make good faith contributions to the open-source world").

177 In a recent work, Thomas Cotter relates meme theory to copyright, and his approach suggests another theoretical lens for indirect voice in my analysis—the message of FOSS licensing as a group of memes, or a memeplex. Thomas F. Cotter, *Memes and Copyright*, 80 TUL. L. REV. 331, 334 (2005) (arguing that the copyright system "impacts not only the quantity of new and distinct memes that are created and published, but also the diffusion, diversity, and quality of the resulting meme pool"). If copyright has this effect, the unique FOSS inversion of copyright can similarly impact meme ecology, and this process would include spreading the FOSS meme—a process I call indirect voice.

178 *See supra* note 5 and accompanying text.

projects. Second, FOSS may enable her transition from neglect to direct voice in some situations. Rather than exit the relationship, FOSS provides an institutional mechanism whereby the employer can offer the employee a higher level of satisfaction. These employment interactions in the Hirschman's framework are related to and embedded in the larger exit and voice interactions between end users and proprietary software providers. The linkage between them further illustrates the importance of this framework in understanding what sustains FOSS and the importance of the FOSS license as an institutional mechanism embodying exit and voice.

## IV
## VOICE FROM THE FOSS COMMUNITY

The preceding three Parts of this Article move progressively from exit to voice. After exploring the nature of FOSS exit for various types of users, and what might chill such exit, the preceding Parts discuss the licensing details governing the character of that exit, and describe the various ways direct and indirect voice attach. The immediately preceding section then examines an embedded subtext, reviewing the influences of exit, voice, and loyalty analyzed in the Hirschman framework as it applied to a moonlighting FOSS contributor.

While indirect voice has already been discussed for the FOSS license, most notably the GPL, and analyzed for the moonlighting FOSS contributor, this Part focuses on other, broader examples of indirect voice from the FOSS community. This indirect voice seeds the greater software ecosystem with the information necessary for a variety of actors to exercise direct voice. This Part examines indirect voice in FOSS activism, license enforcement, and lobbying. These efforts are aimed at exit; they seek to promote a novel way to license, and thus develop and distribute, software. Although designed to promote FOSS exit, these efforts in turn are amplified by such exit when it occurs. There is a synergistic cycle, with each mechanism seeding and promoting the other. As FOSS exit increases, so does the background chorus echoing the indirect voice in the activities below.

### *A. Norm Entrepreneurship and Public Advocacy*

In the age of digital media, FOSS advocacy employs an effective blend of the old and the new. A variety of public figures, many

aligned with one of the two "camps" discussed above, promote various subtexts of the FOSS message. These public figures have a digital presence, typically through writings published on their web sites. However, many of them also travel and speak to promote FOSS.

One of the two most well-known FOSS public figures is Richard Stallman, author of the GPL, founder of the Free Software Foundation (FSF), and a free software developer of considerable repute.[179] The other is Linus Torvalds, the original developer and ongoing leader of the Linux kernel.[180]

Stallman is affiliated with the free software camp. His advocacy, and the work of the FSF and its affiliates, is perceived as less tolerant of proprietary software compared to the open source camp. The FSF houses important FOSS projects and contains extensive web materials discussing FOSS licensing and philosophy.[181] Stallman travels to a great variety of locations to speak about free software. He delivers his message cleverly, with great conviction and compelling logic. He has been controversial in the sense that he seems always willing to assertively voice his disagreement with perceived mischaracterizations of the free software movement.[182] It has been said of Stallman that "[i]f Richard did not exist, it would have been necessary to invent him."[183] Understood in the Hirschman framework, the inescapable need for a norm entrepreneur of Stallman's skill, stature, and persistence is due to the need for indirect voice to buoy the FOSS movement, especially in its early phases.[184]

---

[179] *See* MOODY, *supra* note 60, at 29-30 (describing Stallman as a "geek Moses bearing the GNU GPL commandments and trying to drag his hacker tribe to the promised land of freedom whether they want to go or not").

[180] *See* TORVALDS & DIAMOND, *supra* note 12, at 235-38 (discussing the fame that accompanied developing the Linux kernel).

[181] *See* Free Software Foundation Home Page, http://www.fsf.org (last visited July 19, 2006).

[182] *See* MOODY, *supra* note 60, at 29-30; TORVALDS & DIAMOND, *supra* note 12, at 194-197.

[183] *Contributors*, in OPEN SOURCES, *supra* note 15, at 269.

[184] *See* David McGowan, *SCO What? Rhetoric, Law and the Future of F/OSS Production* 3 (Univ. of Minn. Law Sch. Legal Studies Research Paper Series, Research Paper No. 04-9, 2004), *available at* http://ssrn.com/abstract=555851. *See also* Dan Hunter, *Culture War*, 83 TEX. L. REV. 1105, 1106 (2005) (arguing that the cultural backlash against intellectual property rights is centered in part on the open source movement); Jennifer M. Urban, *Legal Uncertainty in Free and Open Source Software and the Political Response*, http://www.ssrc.org/wiki/POSA (follow "Legal Uncertainty

Torvalds, whose name and affiliation with the open source camp are well-known, does not travel and speak regularly. He orchestrates the Linux kernel development from his office. His communications and his actions, however, have great import. But he is not the fount of indirect voice for the open source camp, in part because his focus is functional: evolving and improving the Linux kernel.

Several others might qualify for the lead role as a fount of indirect voice for the open source camp, and in reality there is no single such person, but I will use the example of Eric Raymond. His writings have been highly influential, and his web site, at the time of this writing, describes that his primary role is to travel and speak to evangelize FOSS.[185]

The examples of prominent figures in the FOSS movement could continue for many pages. The discussion could also include corporate representatives from firms such as Red Hat and IBM, as well as some prominent law professors.[186] The point is not to enumerate all prominent sources of indirect FOSS voice, but to illustrate that the movement includes this effort, and that it is significant.

The message by advocates within each camp support FOSS generally, but emphasize different aspects. Stallman's advocacy is poignantly political.[187] He presents the message that freedom to share software is an absolute necessity to engender self-determination with one's computing resources.[188] His view envisions this freedom as so important that it requires reversing traditional software licensing practices and generally upending notions of intellectual property protection.[189] As a result, his

---

in Free and Open Source Software and the Political Response" hyperlink) (last visited July 19, 2006).

185 *See* Raymond, *supra* note 54.

186 *See, e.g.*, Clint Boulton, *Free Software Foundation Lawyer Eben Moglen Wants to Wipe Out What He Calls the "Scourge" of Proprietary Software*, SERVERWATCH, May 27, 2005, *available at* 2005 WLNR 8437893 (reporting remarks by Columbia University Law School Professor Eben Moglen); *Stanford Law Professor Raps Patents as Barrier to Innovation*, TECHWEB NEWS, Apr. 7, 2005, *available at* 2005 WLNR 5493753 (reporting Professor Lawrence Lessig's remarks concerning the threat software patents pose to FOSS).

187 *See* MOODY, *supra* note 60, at 29-30 (describing Stallman's work as significant because it "provides an ethical backdrop against which the entire free software and open source story is unfolding").

188 *Id.*

189 *See* ROSEN, *supra* note 114, at 107-09 (describing the objectives of the GPL generally).

advocacy leads with political arguments, although it is not devoid of economic considerations.

Raymond's writings, and I presume that his presentations resonate his writings, emphasize economic and technical advantages he finds in FOSS development and distribution.[190] His pragmatic approach to articulating the benefits of FOSS allows more room for some tolerance of proprietary software. One of Raymond's articles posits criteria where FOSS should flourish, but leaves some space in the software ecosystem for proprietary software.[191] His writings otherwise emphasize how FOSS development has structural advantages that lead to superior software.[192]

The advocacy from both camps in the FOSS movement has the common message of inviting exit to FOSS by proprietary software users, even if the emphasized reasons for doing so are different. Less relevant to this Article, but worth acknowledging, is that this advocacy also helps recruit contributors to FOSS. As discussed in Part III above, in some instances, FOSS contributions are indirect exit when provided by a technologist moonlighting from a proprietary software company. Recruiting volunteer technologists to contribute to FOSS is related to inviting corporate users to switch to FOSS: after the switch, the company may have its technologists spend some work time contributing as an act of self-interest to help the community behind the software flourish.[193]

If FOSS advocacy invites exit, it is important that the exit be viable. The viability of the FOSS exit opportunity suffers if the licenses are compromised. Worse, it may diminish confidence in the FOSS licensing system. Thus, it is rational for the FOSS movement to enforce its licenses. The section below reviews some of the FOSS enforcement efforts and theorizes that these efforts also have indirect voice effects.

---

190 *See* MOODY, *supra* note 60, at 144, 148-55 (describing Raymond's background and philosophies generally).

191 Raymond, *supra* note 63, § 10.

192 *See* MOODY, *supra* note 60, at 148-55.

193 *See* Cara Garretson & John Fontana, *Real Deal*, NETWORK WORLD, July 4, 2005, *available at* 2005 WLNR 10973653 (reporting various in-house technology executives' positive perceptions of FOSS, and quoting one CTO as stating, "I would encourage CIOs, if you're going to start using open source you should start thinking early what you're going to give back . . . It stops working if you don't give back").

### *B. License Enforcement as Advocacy Through Legal Forums*

FOSS license enforcement is an organized effort springing from some of the same groups discussed in the preceding section. Its structured and publicized character adds to its indirect voice effect. This section will mention two FOSS license compliance efforts: one centered in the United States, and the other in Europe. Both are significant in that they have resulted in court cases. Both received significant attention in the press that covers FOSS issues, which includes occasional coverage from the major newspapers. Both show the effectiveness of license enforcement to help support the FOSS distribution system, and generally to provide indirect voice about FOSS.

Stallman's FSF and its affiliates again play a central role. The FSF web site has a page dedicated to license compliance called the Compliance Lab.[194] It notes that it answers licensing questions from the community, encourages those questions, offers its services as a paid consultant, and "provides a general 'knowledge infrastructure' concerning the GNU GPL and Free Software licensing."[195]

A linked page, entitled "Negotiating Compliance" discusses the enforcement process when a violation is confirmed, with the FSF noting that "the use of the word 'negotiating' in no way means that the FSF will compromise the principles on which it is organized, namely the necessity of creating and keeping software free."[196] The FSF explains that in most cases a quiet contact resolves the problem, either because the respondent was not aware of the violation, or because she thought she was in compliance and realized that she was incorrect. In recalcitrant cases, the FSF notes that it "has access to the expert legal counsel and the legal resources of the Software Freedom Law Center."[197]

One of the free software camp's most important advocates, Professor Eben Moglen of Columbia University Law School, is chairman of the Board of Directors of the Software Freedom Law

---

194 Free Software Foundation, Compliance Lab, http://www.fsf.org/licensing/compliance.html (last visited July 19, 2006) ("The Compliance Lab has been an informal activity of the Free Software Foundation since 1992 and was formalized in late 2003. The Compliance Lab is our department handling the investigation of the GPL (and LGPL) violations and subsequent enforcement when violations are confirmed.").

195 *Id.*

196 Free Software Foundation, Negotiating Compliance, http://www.fsf.org/licensing/dealt.html (last visited July 19, 2006).

197 *Id.*

Center, as well as general counsel for the FSF.[198] In his capacity with the FSF, he has been involved in its compliance efforts, and his report as an expert is part of the public record of one of the most well-known GPL compliance cases.[199] The case involved the dual licensor MySQL. One of its business affiliates, Progress Software, distributed MySQL's GPL licensed software intermixed with its complementary database software.[200] However, Progress did not supply the source code for its component, an alleged GPL violation because the two software components were intermixed in such a way that the GPL's terms could be required to apply to the Progress component.[201] Professor Moglen's expert declaration provided the analysis explaining how this software interrelationship generated the alleged violation.[202] The case eventually settled[203] after producing only a short district court opinion that mentioned the GPL without deep analysis of the license.[204]

Even though some hoped that the case would produce more court discussion about the GPL, from a license compliance and indirect voice perspective, the case succeeded in requiring Progress to comply with the GPL by releasing the source code.[205] The press coverage and Internet publicity for the case was far beyond what is normal for a licensing dispute between two relatively small suppliers in a niche market.[206] The case drew press coverage because FOSS

---

198 Software Freedom Law Center, Directors, http://www.softwarefreedom.org/team.html (last visited July 19, 2006).

199 Moglen Declaration, *supra* note 56.

200 See Greg R. Vetter, "*Infectious" Open Source Software: Spreading Incentives or Promoting Resistance?*, 36 RUTGERS L.J. 53, 129-30 (2005), for a brief history of the MySQL litigation.

201 *Id.*

202 Moglen Declaration, *supra* note 56, at 7-11.

203 Peter Brown, *Beyond* SCO v. IBM*: Other Legal Issues in the Open Source Community*, 808 PRACTISING L. INST. 103, 112 (2004) (describing the Progress Software case and its settlement).

204 Progress Software Corp. v. MySQL AB, 195 F. Supp. 2d 328, 329 (D. Mass. 2002) (order, at just over one page, granting in part and denying in part MySQL's motion for preliminary injunction).

205 *Id.*

206 *See* Henry W. (Hank) Jones III, *How a Poor Contract Sunk an Open-Source Deal*, LINUX J., Aug. 1, 2002, *available at* http://www.linuxjournal.com/ article.php?sid=6025 (noting that, for a time, many described the case as the "first litigation testing the validity and enforceability of the General Public License" and attributing the parties' dispute to a poorly implemented collaboration agreement). Further, the author posited that the "judge found the GPL issue too uncertain to adjudicate in [the] litigation's early, [preliminary injunction] phase." *Id.*

philosophy, as embodied in the license, was at issue.[207] Also, the general view seemed to be that the case might provide the first "test" for the GPL in a court. This possibility likely heightened interest.

The FSF license compliance program colors FOSS licensing industry-wide. The program goes beyond GPL enforcement to license commentary that discusses the FSF's assessment whether other licenses are compatible with the GPL.[208] These determinations indirectly promote the GPL and stand as a premonition. When GPL-licensed software is mixed with software licensed under other terms, the GPL's terms implicate the mixed software and may require compliance with the GPL, an effect that is popularly known as "viral."[209] If the other software is non-GPL, it might be under a FOSS license compatible with the GPL, in which case there is no compliance problem. Intermixing with a noncompliant license could lead to a license enforcement inquiry from the FSF.

The compatible license analysis is more necessary with the GPL than other FOSS licenses because it is the most widely used license, and among widely used licenses, it has the strongest infectious scope. "Infectious" is the label I used in an earlier article to specify the GPL's conditions requiring that its terms extend to other software when that other software, combined with the GPL-licensed code, creates a derivative work in a copyright sense.[210]

The Hirschman exit and voice framework offers another explanation for the purpose of the GPL's strong infectious scope: it heightens the indirect voice-carrying capacity of the license. Because modern computing uses a layered model for software functionality,[211] the GPL's terms might touch a wide variety of other software, depending on the applications in question. The possibility of the GPL license touching a wide variety of other software increases the incentive of proprietary software license holders to learn about the GPL. This mixes with the tendency of some to view the GPL's infectious terms as expansive, and the resulting notoriety further heightens the indirect voice effect.

---

207 *Id.*

208 Free Software Foundation, *supra* note 44, at GPL-Compatible, Free Software Licenses.

209 *See Copyleft: Is Copyleft "Viral"?*, WIKIPEDIA, http://en.wikipedia.org/wiki/Copyleft (last visited Aug. 14, 2006).

210 Vetter, *supra* note 200, at 58 n.9, 65-66.

211 *See* Robert Plotkin, *Computer Programming and the Automation of Invention: A Case for Software Patent Reform*, 2003 UCLA J.L. & TECH. 7, 38.

The FSF's license compliance program is GPL centered. It ranges from license categorization for GPL compliance to specific investigations and compliance enforcement assistance. The second GPL license enforcement situation arose in Europe.

A Linux kernel developer located in Germany worked on software related to the firewall subsystem in the kernel.[212] Besides its security value to a Linux-based operating system, this code was also valuable to network hardware manufacturers. These manufacturers could use the GPL licensed code to add security and other capabilities. Some did so, but did not release the source code even after distributing the executable software by selling their hardware products, network switches, and other devices.[213] The original developer held sufficient copyright interest in the code base that he was able to bring suit in a German court against the manufacturers. In parallel with the suit, he launched a web site to chronicle the progress of the enforcement effort.[214]

This effort resulted in the first court case upholding the GPL. A German court found that the GPL restrained the switch manufacturers and required them to release the source code as a condition to the continued use of the software.[215] Approximately a year later, the developer was successful against another company in

---

[212] *See* Gnumonks.org, LaForge's Homepage, http://gnumonks.org/users/laforge (last visited July 19, 2006) (noting that the developer, Harald Welte, contributes to a project called "netfilter/iptables, [which] is the firewalling subsystem of the Linux 2.4.x kernel").

[213] Although it is a stereotype, as a general matter, hardware manufacturers are not inclined to release low-level code and commands for controlling their devices. This attitude undoubtedly led the switch manufacturers to use the GPL-protected code, but not disclose the source code modifications they implemented to operate the code on their proprietary hardware.

[214] Gpl-violations.org Project, About the Gpl-violations.org Project, http://www.gpl-violations.org/about.html (last visited July 19, 2006).

[215] Landgericht Muenchen [LG] [Munich District Court] Apr. 2, 2004, Welte v. Sitecom Deutschland GmbH, No. 21 O 6123/04, *unofficial translation available at* http://www.jbb.de/judgment_dc_munich_gpl.pdf (holding by the District Court of Munich that even without a formal agreement, Sitecom cannot distribute the software without making the source code available); *see also* David Graber, *German Court Enjoins Developer for Failure to Comply with GNU License*, 9 ELEC. COM. & L. REP. (BNA) 410 (Apr. 28, 2004) (reporting that the preliminary injunction against Sitecom "follows a series of recent out-of-court settlement agreements between the Netfilter Project and firms making use of the code, including Fujitsu Siemens Computers GmbH and Allnet GmbH"). The court also noted that distribution without source code availability violated Welte's moral rights under German copyright law. *See* Welte v. Sitecom Deutschland GmbH. *See also* Vetter, *supra* note 7, at 670-84 (discussing the relationship between FOSS licensing and the civil law copyright system of moral rights that attach to creative works, including the rights of attribution and integrity).

the German courts, and the developer has achieved significant enforcement success with many other companies without going to court.[216] The developer's web site presents the entire progression of this enforcement effort,[217] and the press noticed and reported these cases.[218]

These ancillary effects of the enforcement results amplify the indirect voice inherent in the compliance program. Not only does the software ecosystem learn about FOSS as the cases receive publicity, but the technical public learns that the important license attributes of the FOSS exit opportunity can be preserved. Before this first German court case, it was common to see in the press and literature devoted to FOSS the indication that the GPL had not been upheld by a court.[219]

The general public advocacy and license compliance efforts discussed above create welcome locales for FOSS and enforce the sanctity of those locales. If both efforts are effective, they buoy the FOSS exit opportunity. However, another force can chill the desirability of that exit: the risk of intellectual property infringement. Both copyright and patent infringement issues concern potential FOSS users. The SCO litigation highlighted the copyright issues.[220] The software patent issues spring from differences in intellectual property protection under patent law compared to copyright. The patent threat provided a very visible forum for indirect voice about FOSS during the debate over the issue of software patents that arose in the legislative process for the European Union's directive to harmonize certain aspects of its patent law.

### *C. Lobbying and the European Union Software Patent Debate*

FOSS indirect voice reached an apex to protect its exit option from the perceived threat of the European Union's directive to harmonize, and allegedly strengthen, certain aspects of its patent law

---

216 *See* David Graber, *German Court Enjoins Software Firm for Failure to Comply with GPL License*, 10 ELEC. COM. & L. REP. (BNA) 417 (Apr. 20, 2005) (reporting that the developer "has negotiated more than [thirty] out-of-court settlements over the past [fifteen] months").

217 Gpl-violations.org. Project, *supra* note 214.

218 *See supra* notes 215-17.

219 *See* Jones, *supra* note 206.

220 *See* SCO Complaint, *supra* note 96; IBM Answer, *supra* note 96.

related to software.[221] While there are perhaps many reasons why the FOSS community would rally against the EU patent directive, I will focus on the inference that stronger patent protection for software has the potential to limit the viability of the FOSS exit option. The FOSS community's analysis of the threat led to an impressive effort against the EU directive. It engendered tremendous indirect voice, implemented by activism of varying degrees including formal lobbying. This activism served two functions. First, it repelled the threat. Second, it helped spread the message about the FOSS exit option.

The patent law threat arises because patent law provides a separate source of intellectual property protection that can attach to software. FOSS uses a licensing system that starts with copyright. The copyright-based terms in a FOSS license create a zone of functional freedom with the software, if one observes the license's conditions. In general terms, as long as contributions to FOSS projects are well-vetted and do not impinge on a third party's copyright, FOSS licensing practices can clear rights within this zone. No third party should be able to use copyright to win an intellectual property infringement suit against the FOSS project if the developer contributions are all truly their original work.[222] Furthermore, in copyright, independent development is a defense. Thus, even if FOSS code is substantially similar to a hostile third party's code, true independent development is a defense to an infringement action.[223]

---

221 *See Commission Proposal for a Directive of the European Parliament and the Council on the Patentability of Computer-Implemented Inventions*, at 2, COM (2002) 92 final (Feb. 20, 2002), *available at* http://eur-lex.europa.eu/LexUriServ/site/en/com/2002/com2002_0092en01.pdf [hereinafter *Commission Proposal on Patentability*].

222 Besides depending on original code contributions, the analysis that FOSS licensing effectively clears a zone for functional freedom also requires that copyright's derivative work right not be infringed. Thus, a developer may put a project at risk by submitting her original contributions along with proprietary modified code from a third party, where the modifications are intermixed with her contributions. This contribution will likely allow the third party to bring an infringement suit based on the derivative work right available under U.S. copyright.

223 Generally stated, "substantial similarity" is a copyright law rubric that forms the basis for an infringement action when copies are not identical. Making an identical copy infringes the reproduction right, unless a copyright defense, such as fair use, is available. Making a substantially similar version of the original also may infringe. The most basic right of copyright, the reproduction right, allows one to exclude others from making copies when no copyright defense exists. Liability would attach in the case where copies are identical, but might also attach when copies are substantially similar.

While the copyright terms in a FOSS license and smart practices to screen code contributions can relieve a project from much of its copyright infringement risk, managing patent infringement risk is trickier because patents can arise independently to threaten a FOSS project. Patent law essentially disregards independent development that arrives after a patent's effective date. A FOSS project takes little comfort against the patent threat by separately conceiving and implementing a method already patented by another. Thus, due to the way patent law works, FOSS developers rightly worry about an unknown patent inhibiting distribution of the software.

The risk of patent infringement is the inhibiting force. I will sketch how this works at a high level, drawing the sketch against the U.S. patent system. To threaten a FOSS technology, a patent must be valid. Specifically, the claims asserted must be valid. A patent is a document that ends with claims, numbered statements describing the product or process of the invention using a range of styles, conventions, and levels of detail. The claims define the scope of the holder's right to exclude. Patent holders write a variety of claims to describe differing facets of the invention and obtain varying breadth of claim scope. Broad claims are easier for defendants to invalidate, but narrow claims in the same patent can remain valid even if the broad claims are invalidated. Anyone who operates without permission, in the scope of a single valid claim, infringes the patent.[224]

In the United States, a patent claim must meet five criteria to be valid; the U.S. Patent and Trademark Office (USPTO) initially evaluates the claim during the patent application process, although the claim can be reevaluated later by a court during an infringement suit. Two of the five criteria, novelty and non-obvious subject matter, are tests that compare the claim to the prior art.[225] The other three, eligible subject matter (sometimes called statutory subject matter), utility, and disclosure, are also measured against the claim.[226] All five must be satisfied for the claim to be valid.[227] The prior art validity requirements implement the commonsense notion

---

[224] 35 U.S.C. § 271(a) (2000 & Supp. 2001-2003) ("[W]hoever without authority makes, uses, offers to sell, or sells any patented invention, within the United States or imports into the United States any patented invention during the term of the patent therefor, infringes the patent.").

[225] *Id.* §§ 102-103.

[226] *Id.* §§ 101, 112.

[227] *Id.* §§ 101-103, 112.

that the patent systems of the world should not grant exclusive rights for a technology that already exists (novelty), or for trivial variations of existing technology (nonobviousness).[228] The paradigmatic example of prior art is an existing patent document that discloses a technology.[229] Another example would be FOSS published on a public web site.[230]

When a patent holder brings a patent infringement suit, she puts the patent at risk in the sense that the defendant might discover "new" prior art that invalidates the claims—"new" meaning that the USPTO did not originally find the prior art. Nonetheless, the threat of a patent infringement suit is a very difficult situation to face because invalidating patent claims, if possible, can be expensive. Patents that have survived invalidity challenges in litigation are the most feared. If the claim(s) of such a patent cover a FOSS technology, that patent holder has great leverage over the creators, distributors, and users of the FOSS.[231]

Even patents not yet tested in litigation provide the holder with significant leverage due to the legal cost necessary to defend an infringement suit. If the holder wins the case, she is entitled to damages that can accumulate rapidly. Moreover, willful infringement allows for up to treble damages.[232] The defendant faced with a variety of broad patent claims is in a tough spot. Even if she invalidates the broad claims by discovering new prior art, her FOSS technology might still fit within the language of one of the narrow claims, leading to infringement.

These functional realities about the patent system mix with the following historical fact to frame the EU patent proposal debate: in the late 1990s, case law in the United States expanded the scope of eligible subject matter to include methods implemented by pure

---

228 *Id.* § 103.

229 *Id.* § 102(a)-(b); *see e.g.*, U.S. Patent No. 6,738,905 (filed May 18, 2004) (an example of a patent that would serve as prior art).

230 *See* Stephen M. McJohn, *The Paradoxes of Free Software*, 9 GEO. MASON L. REV. 25, 50-52 (2000) (arguing that systemically, and over the long run, FOSS may inhibit patent infringement suits because open and available source code increases the chance that litigants will discover patent-invalidating prior art).

231 *See* ROSEN, *supra* note 114, at 135-36, 289-90.

232 *See* 35 U.S.C. § 284 ("Upon finding for the claimant the court shall award the claimant damages adequate to compensate for the infringement[;] . . . the court may increase the damages up to three times the amount found or assessed.").

software.[233] Eligible subject matter is the first of the five validity criteria, and in this context, the question of statutory interpretation is whether such methods are a "process."[234] While the domain of patents is very broad, some subjects are not eligible for patent protection. More precisely, there are a small number of narrow exceptions from the very broad domain of patent-eligible subject matter. The United States Supreme Court has acknowledged these exceptions by remarking that the formula $E=mc^2$ is ineligible subject matter as a law of nature.[235] Another exception is for "abstract algorithms."[236] The United States case law revisions in the late 1990s eliminated a long-simmering doubt about the patentability of methods implemented in software.[237] This doubt was at its apex when the claim(s) did not recite computing infrastructure and tie the method to physical structure. With the doubt extinguished, the floodgates opened in an upswing of United States patents for methods implemented in software. This flood of software patents creates a generally higher risk of patent infringement for FOSS, as well as all licensed software in general. It also allows FOSS competitors, such as Microsoft, to wield patents competitively against FOSS if they so desired.[238]

---

233 *See* State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1375 (Fed. Cir. 1998). The patent at issue in *State Street* claimed a computer system that calculated asset values for a particular configuration of entities sharing participation in pooled mutual funds. *Id.* at 1371; AT&T Corp. v. Excel Commc'ns, Inc., 172 F.3d 1352, 1353-54, 1360-61 (Fed. Cir. 1999) (extending the holding of *State Street* to a pure process claim for a long-distance messaging technique to facilitate charge billing).

234 35 U.S.C. § 101 ("Whoever invents or discovers any new and useful *process*, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefore . . . .") (emphasis added).

235 Diamond v. Chakrabarty, 447 U.S. 303, 309 (1980) (noting that "[t]he laws of nature, physical phenomena, and abstract ideas have been held not patentable") (citations omitted). Even if the formula was eligible, you could not patent it today because it is in the prior art.

236 Diamond v. Diehr, 450 U.S. 175, 185-86 (1981) (noting that statutory subject matter does not include unapplied, abstract mathematical formulas or algorithms, the latter defined as a "procedure for solving a given type of mathematical problem") (quoting Gottschalk v. Benson, 409 U.S. 63, 65 (1972)).

237 *See* Julie E. Cohen & Mark A. Lemley, *Patent Scope and Innovation in the Software Industry*, 89 CAL. L. REV. 1, 11-14 (2001).

238 *See* Daniel Lyons, *Linux Scare Tactics*, FORBES.COM, Aug. 2, 2004, http://www.forbes.com/home/enterprisetech/2004/08/02/cz_dl_0802linux.html (reporting that fears regarding patent suits against Linux users are on the rise, but may be the result of suggestions by those hoping to cash in by offering insurance).

From the perspective of the FOSS community, its opposition to the EU patent proposal arose from the concern that the EU proposal would cause an increase in software patents issued by European nations similar to the increase that followed the case law changes in the United States. In addition, the FOSS community was concerned that the European patent law would follow the United States in allowing broader patent claims that would create a greater threat of infringement, assuming validity. The proposal also became a forum to debate software patents generally.

The situation in Europe before the proposal was somewhat ambiguous. EU member states grant patents individually. In parallel, and outside the purview of the EU, many European nations are members of the European Patent Convention, a treaty-based organization that established the European Patent Office (EPO), allowing inventors to file in a central location and streamline the process to obtain patents in multiple European nations.[239] Europe's federated system of patent protection resulted in differing interpretations at the national level, principally in Germany and the United Kingdom, as to whether methods implemented in software were eligible subject matter.[240]

Given the uncertain status of software patents, the EU proposed a directive to harmonize member states' administrative procedures and local law. Its self-described goal is given in the quote below:

> [T]he legal rules as interpreted by Member States' courts should be harmonised and the law governing the patentability of computer-implemented inventions should be made transparent. The resulting legal certainty should enable enterprises to derive the maximum advantage from patents for computer-implemented inventions and provide an incentive for investment and innovation.[241]

A directive is a binding command to the member states, but each nation must transpose the directive into its local law, typically with legislation at the member state level. The EU Software Patent Directive ("Directive") never made it that far. The complex,

---

[239] *See* http://www.european-patent-office.org/epo_general.htm (last visited July 21, 2006) (noting that the EPO was "[e]stablished by the Convention on the Grant of European Patents (EPC) signed in Munich 1973, [and that] the EPO is the outcome of the European countries' collective political determination to establish a uniform patent system in Europe").

[240] *Commission Proposal on Patentability*, *supra* note 221, at 2.

[241] *Id.* at 17-18.

multistage, EU "codecision" legislative process extended over many years, but the record shows FOSS community impact at each step of the way.

Oversimplifying, the EU Parliament consistently insisted on Directive amendments to aid FOSS, but these amendments were not acceptable to the Council and the Commission, the other two "branches" of the EU.[242] Eventually recognizing an impasse, the Directive died, preserving the status quo so that neither side could gain advantage through the Directive.[243] I will not review the entire legislative process, or try to trace the proposed amendments at each stage. The important point is to note the FOSS community voice in the process.

Issued in 2002, the original Directive text acknowledged legislative process input by the FOSS community and industry:

> The individual responses were dominated by supporters of open source software, whose views ranged from wanting no patents for software at all to the "official" position of the Eurolinux Alliance which is to oppose patents for software running on general-purpose computers. . . . [A]lthough the responses in [the industry] category were numerically much fewer [than] those supporting the open source approach, there seems little doubt that the balance of economic weight taking into account total jobs and investment involved is in favour of harmonisation along the lines suggested . . . .[244]

In the first pass through the EU Parliament, its members proposed a variety of amendments to the Directive, acknowledging input from the FOSS community.[245] These revisions triggered procedures

---

242 *See generally* Robert Bray, *The European Union "Software Patents" Directive: What Is It? Why Is It? Where Are We Now?*, 2005 DUKE L. & TECH. REV. 11 (profiling changes proposed to the Directive).

243 Nikki Tait, *European Position Is Left Patently Unclear*, FIN. TIMES, Sept. 19, 2005, *available at* 2005 WLNR 14734897.

244 *Commission Proposal on Patentability*, *supra* note 221, at 4.

245 European Parliament Committee on Legal Affairs and the Internal Market, *Report on the Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-Implemented Inventions*, 20, COM(2002) 92—C5-0082/2002—2002/0047(COD) (June 18, 2003), *available at* http://www2.europarl.eu.int/omk/sipade2?PUBREF=-//EP// NONSGML+REPORT+A5-2003-0238+0+DOC+PDF+V0//EN&L=EN&LEVEL =2&NAV=S&LSTDOC=Y ("The rapporteur has also carefully weighed the arguments put forward by industry and the open source community.").

The rapporteur also reported "being harassed by lobbyists." Minutes, Proceedings of the Sitting, 2004 O.J. (C 77) 18, 19, *available at* http://europa.eu.int/eur-lex/lex/LexUriServ/site/en/oj/2004/ce077/ ce07720040326en00180019.pdf.

whereby the EU Council eventually adopted a proposal that reverted to the original approach and transmitted it to the EU Parliament in 2005, where the Parliament once again proposed amendments.[246] This second batch of amendments caused the Directive-killing impasse. They included requirements to monitor the effects of the Directive on the open source software community and ensure the availability of patents on a reasonable and nondiscriminatory royalty basis for interoperability and licensing in the public interest.[247] The explanatory statement accompanying the amendments makes clear that the EU Parliament disagreed with the Council and the Commission as to the impact of the Directive's approach, and wanted to make sure that the Directive did not expand software patent availability in Europe.[248] Lobbying by FOSS advocates facilitated the Parliament's understanding of the threat increased software patenting poses to FOSS.[249] This indirect voice helped stalemate the political process over the Directive, preserving the status quo.[250]

Thus, besides repelling the threat of expanded software patenting in Europe, the lobbying effort against the EU Directive is similar to the general FOSS activism and license compliance efforts in that it illustrates indirect voice within the Hirschman framework. All of

---

246 *See* European Parliament Committee on Legal Affairs, *Recommendation for Second Reading on the Council Common Position for Adopting a Directive of the European Parliament and of the Council on the Patentability of Computer-Implemented Inventions*, 4-22, 11979/1/2004—C6-0058/2005—2002/0047(COD), *available at* http://www2.europarl.eu.int/omk/sipade2?PUBREF=-//EP// NONSGML+REPORT+A6-2005-0207+0+DOC+PDF+V0//EN&L=EN&LEVEL =2&NAV=S&LSTDOC=Y.

247 *See id*. at 14.

248 *See id.* at 22-24.

249 *See, e.g.*, Bray, *supra* note 242, ¶ 21 (noting that the first reading of the proposal occurred "against the background of fierce and unconventional, but extremely effective, lobbying by the open source community"); Free Software Foundation Europe, Software Patents in Europe: Memorandum on Software Patentability, http://www.fsfeurope.org/projects/swpat/memorandum.en.html (last visited July 21, 2006) (arguing that "the Council of the European Union frustrated . . . democratically-reached [anti-software-patent] positions—they restored the original proposal with unlimited patentability of software"); *Open Source Leaders Slam Patents*, BBC NEWS, Feb. 3, 2005, http://news.bbc.co.uk/1/hi/technology/ 4229689.stm (reporting that Linus Torvalds stated that software patents were a problem for FOSS).

250 *See* News, Marks & Clerk Patent and Trade Mark Attorneys, The Software Patents Directive Is Dead—Long Live Software Patents! (July 6, 2005), http://www.marks-clerk.com/attorneys/news_one.aspx?newsid=55 (noting that while "[t]he rejection means that there is no formal harmonisation across the European Union . . . it also means that patent protection is at least no worse than it has been: current practices of both the European Patent Office and the EU member states are maintained").

these efforts provide a background signal from which direct voice takes form, allowing software users either to threaten exit to FOSS, or in aggregate (but without necessarily coordinating their activities), to impart a disciplining force that operates along many dimensions of the proprietary software licensing model.

## V
## EXIT AND VOICE IMPLICATIONS FOR FOSS LICENSING

As Parts I through IV demonstrate, FOSS is laced with mechanisms of exit and voice that converge on the software user. This framework anchors a new perspective for the interactions and influences channeling FOSS at a critical juncture: the user adoption decision. Joining other explanations for the FOSS phenomenon, Hirschman's framework enables a better understanding of the movement and the licensing that underlies it.

Within each situation discussed above, the analysis uncovers the means by which exit and voice cooperatively, or individually, influence an important aspect of FOSS. In each context, the corresponding part catalogs the implications from the Hirschman perspective. This Part's purpose is to show some overarching implications with three tentative conclusions. First, this framework adds support to the thesis put forth elsewhere that FOSS generation and adoption will continue to be most successful for platform software technologies, especially where market leveraging behavior by incumbent firms triggers antimonopoly passions. Second, in the competition among licenses for future mindshare and codeshare, licenses with greater synergy between exit and voice may continue to have an advantage. Third, courts should consider the Hirschman framework when evaluating legal issues related to FOSS licensing.

### *A. The Exit and Voice Framework May Channel FOSS to Platform Applications*

Certain scholarship tries to describe the class of applications for which FOSS will emerge or be successful. Some of this work suggests that FOSS licensing is more effective when used for platform technology,[251] such as operating systems, Internet

---

[251] *See* Raymond, *supra* note 63, § 19 (discussing a high payoff for use of open source in an application that "establishes or enables a common computing and communications infrastructure").

"middleware," and network protocols. This may spring from notions of complementary economics.[252] A platform technology can enable a wider range of complements. If economic activity with those complements can generate spillover or contributions back to the platform, a FOSS-enabling cycle might result.[253] Another theory focuses on the development process, specifically the benefits of source code availability. Under this logic, FOSS is most conducive to platform applications because these have the greatest possibility for "massive peer review"—a euphemism for the scrutiny a large developer and user base can supply to the internal workings of FOSS.[254]

The exit and voice framework supports these arguments, adding a new perspective as to why FOSS is likely to find success with platform applications. Certain software application classes may generate pent-up voice, either because the dominant products are from a company with market leverage or power, such as Microsoft, or because the applications have a personal milieu, such as e-mail or web browsing. FOSS exit may release the pent-up voice in direct or indirect form.

During the early 2000s, FOSS, and in particular the GNU/Linux operating system, became known as the biggest threat to Microsoft.[255] During this time, the United States federal courts also adjudged Microsoft to be a monopoly in certain markets.[256] The FOSS movement is too complex to characterize as a mere Microsoft

---

252 *See* Joel West, *How Open Is Open Enough? Melding Proprietary and Open Source Platform Strategies*, 32 RES. POL'Y 1259, 1259-66 (2003).

253 *See id.*

254 *See* Raymond, *supra* note 63, §§ 11, 13-14, 16.

255 The press reports characterizing FOSS as Microsoft's biggest threat are too innumerable to catalog. The recognition, however, goes a step further to Microsoft itself, who both in the press and in its Securities and Exchange Commission (SEC) filings has acknowledged the threat. Microsoft Corp., Quarterly Report (Form 10-Q), at 31 (Apr. 22, 2005), *available at* http://www.shareholder.com/visitors/ActiveEdgarDoc.cfm?id=3633254&f=rtf ("[T]he popularization of the non-commercial software model continues to pose a significant challenge to our business model. . . . To the extent open source software gains increasing market acceptance, sales of our products may decline, we may have to reduce the prices we charge for our products, and revenue and operating margins may consequently decline."). *See also* Steve Lohr, *Pursuing Growth, Microsoft Steps Up Patent Chase*, N.Y. TIMES, July 30, 2004, at C3 (suggesting that Microsoft's plans to increase patent filings were in response to growing competition from open source products).

256 *See* United States v. Microsoft Corp., 253 F.3d 34, 50-78 (D.C. Cir. 2001) (affirming the lower court's finding of monopolization of the personal desktop operating system market).

backlash. The mechanisms of exit and voice, however, may be uniquely concentrated against a monopoly. Although exit from a monopoly may not be available, there is a desire for exit if product or service quality or price, that is, overall value, is or becomes dissatisfying or less satisfying.[257] When exit is not fully available, voice directed to a monopoly is an option, but it may or may not be effective.

In the United States, there has traditionally been a stigma against monopoly.[258] At times, United States law and society was more inclined to police monopolies strictly.[259] An antimonopoly bias can heighten the intensity of voice directed to monopolies or near-monopolies, such as Microsoft. Thus, when a viable exit opportunity arises, this generates publicity. This publicity is the pent up, unsatisfied voice that now has something to talk about: an exit possibility.

The switching costs and network effects that produce user lock-in to the Windows operating system provided Microsoft the opportunity to leverage its position.[260] Its leveraging generated voice among its users and the public at large as the significance of Windows grew with the explosion in desktop computing and the Internet. In the original Hirschman framework, voice is the only mechanism against a monopoly because customers have no alternatives.[261]

Microsoft's Windows monopoly, however, was never complete. There were always alternatives, such as Unix or Apple, but exit to those systems was less viable than exit to GNU/Linux. Apple only has a market niche because its operating system only runs on its hardware. The Unix systems are expensive compared to the lower-level computers used for Windows. Moreover, in comparison to GNU/Linux, and to Windows, the Unix systems charged expensive software royalties.[262] GNU/Linux, in comparison, while not a

---

257 *See* HIRSCHMAN, *supra* note 1, at 26-27, 55-61.

258 *See* HERBERT HOVENKAMP, FEDERAL ANTITRUST POLICY: THE LAW OF COMPETITION AND ITS PRACTICE § 2.1-2.2 (3d ed. 2005) (reviewing the history and ideology of American antitrust policy, notably the 1950s to 1960s Warren Era that was "openly hostile toward innovation and large scale development").

259 *Id.*

260 *See* Ferguson, *supra* note 88, at 66.

261 HIRSCHMAN, *supra* note 1, at 33, 55.

262 *See* WEBER, *supra* note 37, at 39 (noting that Unix royalty rates in the late 1980s and early 1990s surpassed $100,000).

perfect substitute for Windows, had one critical attribute that earlier exit options did not: it runs on virtually all the same personal computer hardware that supported Windows.

The FOSS exit from Microsoft is often perceived as attractive economically, but it may also scratch a political itch for many users. It is generally accepted that, at the time of this writing, the GNU/Linux operating system is better suited for technically sophisticated users.[263] This eliminates many potential switching users, mostly nontechnical personal users, but leaves a great mass of institutional users and tech-savvy personal users with a feasible option to exit. The institutional users purchase in volume, so their potential exit is of particular note to proprietary software providers such as Microsoft. In one example of such an exit, the city of Munich, Germany chose a GNU/Linux approach for its many thousands of computers despite fierce competition by Microsoft for the business.[264] This transaction was widely reported in the press, and the voice tingeing this exit had a dimension of international political intrigue: Munich exited Microsoft's Windows product[265] at a time when the European Union competition authorities were adjudicating an enforcement action against Microsoft.[266]

Other political itches related to software might come from personal applications: the computing tasks that occupy almost all developed-country users, such as e-mail, web browsing, and creating or using documents, presentations, or data. These applications are political in the sense that their ubiquity creates the likelihood for voice when dissatisfaction occurs. The most poignant example is

---

263 Stephen K. Kwan & Joel West, *A Conceptual Model for Enterprise Adoption of Open Source Software*, *in* THE STANDARDS EDGE: OPEN SEASON (Sherrie Bolin ed., 2005, forthcoming) (manuscript at 7, *available at* http://www.cob.sjsu.edu/OpenSource/Research/KwanWest2005.pdf).

264 *Technology Briefing Software: Microsoft Loses Munich Contract to Linux*, N.Y. TIMES, May 29, 2003, at C6 (reporting that Munich officials planned to switch over 14,000 computers to the Linux operating system, and quoting Munich's Mayor as saying that the city's decision "doesn't just ensure more provider independence for its I.T. infrastructure, but also sets a signal for more competition in the software market").

265 Munich's exit was not a complete exit. It will require a lengthy process of several years to switch over from the Windows computers to the GNU/Linux systems. In that sense, Munich will continue as a Microsoft customer for many years. Moreover, it may continue to use other Microsoft products apart from the reported project.

266 *See* Commission Decision (EC) No. C(2004)900 final of 24 Mar. 2004, *available at* http://europa.eu.int/comm/competition/antitrust/cases/decisions/37792/ en.pdf (finding that a protocol to authenticate users for purposes of granting access to computing resources had been extended beyond the relevant standard).

political voice from e-mail spam and malware, which resulted in a federal law hoping to curb spam[267] and in a bevy of state bills aimed at malware when the problem became particularly acute in the early 2000s.[268] The potential for voice builds and amplifies when an application is prominent and widely used—its problems are sometimes equally ubiquitous and the populace turns to politicians for solutions.

If a widely used platform application has greater attendant voice potential, this heightens the likelihood of releasing a pent-up response when a FOSS exit becomes available. A FOSS alternative application will allow some users a noisy exit, triggering the perhaps latent or muffled voice from these or other users.[269] Customers who do not switch may feel better because they have a FOSS alternative with its advantages of no royalties and functional freedom. Even if a user is not inclined toward FOSS tenets, FOSS applications can offer her an exit alternative with intriguing characteristics if she becomes dissatisfied with her current application. As a result, the original ubiquitous application provider(s) may respond with improvements, or begin to address the direct voice it might have previously ignored.

Beyond Windows, one could posit a list of platform application classes that may be future FOSS targets. Notables on the list would include Internet browsers,[270] e-mail user interfaces,[271] and relational

---

267 Controlling the Assault of Non-Solicited Pornography and Marketing Act of 2003, Pub. L. No. 108-187, 117 Stat. 2699 (requiring labeling of unsolicited commercial e-mail messages, opt-out instructions, and a physical mailing address of the sender to be included with the messages, while prohibiting the use of deceptive subject lines and false headers).

268 *See* Susan W. Brenner, *State Cybercrime Legislation: Disseminating Viruses and Other Harmful Code*, *in* DATA SECURITY AND PRIVACY LAW: COMBATING CYBERTHREATS § 15:20 (Kevin P. Cronin & Ronald N. Weikers eds., 2005).

269 Hirschman discusses the concept of alert and inert customers and relates the concept to unused voice in reserve. *See* HIRSCHMAN, *supra* note 1, at 24, 32.

270 *See* John Markoff, *Mozilla Plans Faster Growth for Its Browser*, N.Y. TIMES, Aug. 3, 2005, at C5 (reporting that Internet Explorer competitor, Mozilla, recently created a for-profit subsidiary to provide fee-based service and support for its Firefox product, and noting that some estimates place Firefox's market share at approximately ten percent).

271 Novell's recent acquisition of Ximian, Inc. includes collaboration software called Evolution, now pitched as a direct competitor to Microsoft Outlook. *See* William M. Bulkeley, *Novell's Linux Bet Could End Its Losing Streak*, WALL ST. J., Apr. 1, 2004, at B3. *See also* Novell, E-Mail, Calendaring and Collaboration, Novell Evolution 2, http://www.novell.com/products/desktop/features/evolution.html (last visited July 24, 2006) (noting Microsoft Exchange server support is included).

databases.[272] The first two have a personal milieu, and the third is almost as important a platform as the operating system.[273]

In sum, platform software applications have a greater potential to engender voice. The traditional reasons used to differentiate the economic nature of software apply; network economic effects with resulting lock-in phenomena, along with the sometimes high costs and risks associated with switching, give users the impression of being overly tied to a vendor or technology. These effects are heightened when a platform software technology is involved, as opposed to a software product or technology that does not underlie numerous complementary systems. The product or technology at issue is often highly differentiated and complex. As a result, if the product or technology becomes less than fully satisfying, the user's ties to the product may engender voice because exit is not as easy as exiting a commodity product or service.[274]

Compound these effects with one of two influences (the vendor is perceived as monopolistic or the application is highly integrated into the daily life of the citizenry, such as with e-mail), and there is an even greater chance for voice. While exit to FOSS still entails the burdens of switching, the exit may give greater release to the voice because it may be perceived as an exit to an alternative with a lesser degree of restrictive future lock-in effects.[275]

---

[272] *See* Jay Lyman, *Open Source Databases Gaining Ground, Analysts Say*, NEWSFORGE, Apr. 19, 2004, http://software.newsforge.com/article.pl?sid=04/04/14/1347227 (noting that open source database leaders such as MySQL, PostgreSQL, and Berkeley DB continue to gain market shares in both Unix/Linux and Windows environments).

[273] The relational database space, compared to operating systems, is more oligopolistic on both the proprietary and open source side. Thus, there is more opportunity for exit among vendors within the proprietary class of applications. Moreover, as of this writing, the FOSS alternatives are still adding functionality to establish themselves as a true equivalent to enterprise strength relational databases such as the products from Oracle Corporation. To the extent database users migrate to FOSS databases, this may be more of an exit mechanism to reduce software royalty costs than a voice mechanism.

[274] Hirschman's framework acknowledges that "[t]he willingness to develop and use the voice mechanism is reduced by exit, but the ability to use it with effect is increased by it." HIRSCHMAN, *supra* note 1, at 83. Against exit and voice, Hirschman posits loyalty as a mechanism that holds exit at bay, at least for a time, and accordingly activates some voice by otherwise exiting users who presumably are vocal while delaying exit. *Id.* at 78, 83. Such loyalty might be a reasoned, calculated act by users. *Id.* at 79. In the case of software lock-in from network effects and switching costs, the loyalty seems forced. Even so, however, forced loyalty may generate voice as long as some of the forced loyalists tend to be vocal as they remain loyal.

[275] *See* Ferguson, *supra* note 88 (positing that open source "severely limits the possibility of propriety 'lock-in'—where users become hostage to the software vendors

Post-exit, the new FOSS user is locked in to the FOSS development community for the product or technology. This is possibly seen as a lesser evil (or more likely seen as a "good"), however, because most FOSS licensing practices guarantee that the user can practice software self-help if she is sufficiently technically proficient, or try to contribute to the development effort or influence the software using the voice mechanisms of the FOSS community. FOSS voice is thought to offer a greater degree of transparency than the voice mechanisms traditionally employed by proprietary software technologies, even if they do not necessarily guarantee greater effectiveness in terms of implementing a particular user's desires.[276]

Given the potential for greater release and exercise of voice in a platform application, the insights from the Hirschman framework suggest that FOSS applications will be more likely to occur or be successful with platform applications. Upon exit to FOSS in platform applications, the synergistic effects of both exit and voice found in the FOSS license take a greater potency.

---

whose products they buy—and therefore eliminate incentives for vendors to employ the many tricks they traditionally use on each other and on their customers").

[276] The degree of transparency for user voice is likely dependent on the difference between user voice mechanisms for proprietary software versus FOSS. FOSS development frequently shows more transparently what the developing organization does with user feedback, and why it does so. Besides taking customer feedback via sales, marketing, and customer support, technology companies often initiate, facilitate, or support "user group" organizations. Within Hirschman's framework, user groups allow a company to lubricate the voice mechanism. *See* HIRSCHMAN, *supra* note 1, at 42-43. However, in many companies, exit is easier for customers than voice, especially customers of commoditized goods or services. *Id.* at 39-41. Voice requires the user/customer to develop a message and find a way to convey it to the company. *Id.* at 80. To the extent this is more difficult than exit, Hirschman posits that organizations lose the potentially valuable disciplining force of voice. *Id.* Accordingly, a smart organization facilitates voice-providing measures. With proprietary software development, there is an incentive, and a practice, to take a lot of input, and then decide internally what features to implement. TORVALDS & DIAMOND, *supra* note 12, at 229-33. The reasoning behind such decisions is explained at a high level of generality in order to placate the disappointed or minimize disclosures to competitors. FOSS software development could follow a similarly cloistered model. Many projects do not, however, because the development history inherently is open. Any user/developer submitting the actual code for a new capability can share with the rest of the community any reasons for rejection, or the fact of specific rejection. The transparency arises in part from development tools that tend to store all communications about the code and the process in an Internet-accessible, centralized repository. On the other hand, proprietary software vendors often provide similar online resources for non-source-code technical information, and for interacting with developers.

### *B. Exit and Voice as Competitive Assets for FOSS Licenses*

The voice mechanism discussed in Part II is the FOSS license. Both its legally operative language and its ancillary materials implement and carry indirect voice for the particular FOSS tenets expressed in a given license. Its substantive provisions enable the FOSS-licensed software to provide users exit from an equivalent proprietary-licensed application. Thus, many FOSS licenses involve both mechanisms.

A license's indirect voice is not necessarily correlated with the legal effect sought. Licenses of greater legal effect might have minimal precatory voice, as in the case of the corporate-style licenses,[277] or thundering precatory voice, as in the case of the GPL. Since FOSS licenses facilitate user exit from proprietary-licensed software, yet also have a voice delivering potential, this raises the question of the role of this indirect voice in the proliferation or popularity of a particular license. On a broader scale, recognizing voice in licenses within the Hirschman framework queries the role of such voice, in conjunction with exit, in the processes that influence license generation, selection and competition in organized and informal ways.

FOSS licenses proliferated as FOSS gained popularity[278] for at least three reasons. First, the character of the FOSS community inherently suggests writing new licenses. Enterprising programmers who start a FOSS project often care deeply about the terms for sharing and ensuring functional freedom for the software. They are going to write new code, so why not write a new license as well? Second, the OSI indirectly promotes license proliferation through its certification program. The success of the program in attracting licenses for evaluation against the OSD has created an increasing

---

277 *See supra* Part II.A.1.

278 *See* Brian W. Carver, *Share and Share Alike: Understanding and Enforcing Open Source and Free Software Licenses*, 20 BERKELEY TECH. L.J. 443, 457 (2005).

number of OSD-certified licenses.[279] Third, some corporations who release software under FOSS terms write their own licenses.[280]

License proliferation has become a problem for many in both the free software and open source camps of the FOSS community in the mid-2000s.[281] This concern led the OSI to change their certification process by adding three new requirements. New licenses will be approved only if they meet the original criteria and are also "(a) non-duplicative, (b) clear and understandable, and (c) reusable."[282] Moreover, the OSI will classify existing and future licenses into Preferred, Approved, and Deprecated. In essence, the OSI added a new sorting step to its license evaluation.[283]

Viewed through Hirschman's framework, the OSI process shows structural features favoring licenses that facilitate greater exit opportunity. This is particularly the case for the original conditions. Besides the basic FOSS tenets of free redistribution and available source code, the OSD criteria requires that a license not discriminate against persons or groups, fields of endeavor, products, or technologies.[284] These antidiscrimination provisions have the effect of ensuring that approved licenses enable the widest possible use of the software.[285] Thus, FOSS licensed under an OSD-approved license presents an exit opportunity to a wide audience thanks to the antidiscrimination criteria.

The new OSI conditions implement a license hierarchy, which can be hypothesized as a process that isolates and amplifies a few

---

279 OSI established a License Proliferation Committee to discuss the increasing number of licenses in use. Open Source Initiative, *supra* note 118. Among the proposals under consideration, committee members will be assigning current OSI-approved licenses to a tiered structure, where Tier 3 licenses remain approved but no longer recommended for future use. *See* Open Source Initiative, License Proliferation, http://www.opensource.org/docs/policy/ licenseproliferation.php (last visited July 24, 2006) [hereinafter OSI LP].

280 Open Source Initiative, *supra* note 118.

281 *See, e.g.*, OSI LP, *supra* note 279 (highlighting a project to sort licenses into three tiers). *See also supra* note 118 and accompanying text.

282 *I-Technology Viewpoint: Open Source Is Open to Debate*, COLDFUSION DEVELOPER'S J., http://coldfusion.sys-con.com/read/49143_p.htm (last visited Oct. 6, 2006).

283 *See* Open Source Initiative, *supra* note 118.

284 *See* OSD, *supra* note 42.

285 Without the antidiscrimination provisions, one can imagine licenses that prohibit use of the software in activities that a FOSS developer finds disagreeable. An example of a noncompliant term given in the OSD is a license provision prohibiting use of the software "in a business, or from being used for genetic research." *Id.*

licenses from the growing cacophony of licenses submitted for OSD approval. This new structural feature of the OSD gives credence to voice, demanding that licenses be non-duplicative to minimize the cacophony, and "clear and understandable" to amplify, or at least effectively transmit, indirect voice. By elevating some licenses over others, the indirect voice embedded in these licenses takes on greater volume. Both the original OSI conditions as well as the new criteria favor licenses with greater enabling capacity for one mechanism or the other.

The OSI license sorting process can be thought of as a competition among licenses. A similar evaluation process occurs within the entire FOSS community.[286] In either case, the question presents itself as to the degree indirect voice in a license helps its adoption rates or OSI classification success. The most widely adopted FOSS license, the GPL, has a high degree of indirect voice. But this correlation cannot be taken as causation because there are many other explanations for the GPL's adoption success, the most obvious of which is that it was first.[287] On the other hand, there is a sense that the strong indirect voice in the GPL is a part of its adoption success. This has been observed in a different way by David McGowan in noticing that the GPL has a trademark-like effect.[288] To function as a mark or a brand, the GPL embodies views about how software should be handled. There is a baseline that people expect for GPL-licensed software. When programmers select a license for a new FOSS project, they may desire to endorse those baselines and increase the indirect voice in the GPL.

Generalizing and evaluating indirect license voice beyond trademark-like effects, however, raises empirical questions that do not produce simple answers. FOSS licenses serve as informational devices. When a user is familiar with a license's terms, as many are

---

286 *See* Josh Lerner & Jean Tirole, *The Scope of Open Source Licensing*, 21 J. L. ECON. & ORG. 20, 24-31 (2005) (describing attributes of open source licenses that influence their likely use on FOSS projects, and the decision-making process of a potential FOSS user generally).

287 Bert J. Dempsey et al., *Who Is an Open Source Software Developer?*, 45 COMM. ACM 67, 71 (2002) (reporting from a study of contributor submissions to the non-kernel components in the GNU/Linux operating system that over half of the software submissions identify the GPL as the applicable license).

288 McGowan, *supra* note 184, at 33-34 (suggesting that the GPL terms may not necessarily be optimal for the developers who use them, but are employed in a trademark sense as a quasi-brand identity espousing certain development procedures or ideological beliefs developers may find more important than the terms themselves).

with the GPL, she can evaluate the exit opportunity for the GPL-licensed software without necessarily rereading the license. Whatever indirect voice the license triggers may also register with the user. The empirical question concerns the degree to which this indirect voice influences the user's adoption decision for the software.[289] Software functionality is often layered, complex, and hard to quantify. Thus, the adoption evaluation is the result of many factors, the license attributes being a subset of these factors.[290] The substantive license terms are its exit attributes, and those same terms and any precatory language comprise the license's indirect voice. While this Article's scope does not include the empirical inquiry into the degree this indirect voice influences the user's adoption, such an influence might exist, especially in the case of the GPL.

To pose the question of voice influence across all FOSS licenses expands the inquiry. In many cases, the user evaluating the software may not be familiar with the FOSS license's terms. Thus, a second empirical question arises: is the user more likely to read the FOSS license as opposed to reading a proprietary license? One reason to think that the answer is "yes" is due to the reputation of FOSS licenses. During the first half of the 2000s the FOSS movement gained considerable headway. In that time, the legal practice literature and general press coverage of FOSS exploded. Some of these materials cautioned users about the due diligence necessary to use FOSS or incorporate it into an organization's information technology operations.[291] Thus, the potential adopting user may be conditioned to undertake greater due diligence for FOSS licenses. Whether this translates into a greater likelihood to read the licenses is hard to gauge. If proprietary mass-market software licenses for Windows-based software or web sites are rarely read, one reason might be that users have an expectation of their terms: a nonexclusive grant to use the software on only one computer or for only one user. Whether a similar expectation of homogeneity exits for FOSS licenses is also hard to gauge, although there is recognition

---

289 Joel West & Jason Dedrick, The Effect of Computerization Movements upon Organizational Adoption of Open Source 20-23 (Feb. 28, 2005) (unpublished paper, http://www.cob.sjsu.edu/OpenSource/Research/WestDedrick_SI_2005.pdf) (finding little support for the notion that the ideology embodied in the GPL influenced its adoption, but that increased user choice and control did influence adoption).

290 *See* Gomulkiewicz, *supra* note 66, at 898-900.

291 *See generally* Laura A. Majerus, *Avoiding and Curing Open Source Problems*, 808 PRACTISING L. INST. 189 (2004) (describing policies and internal software auditing methods to manage FOSS licensing issues for companies using FOSS).

of two broad classes of licenses: attribution-only licenses and licenses patterned after the GPL, which includes corporate-style licenses in this Article's license categorization.

On a continuing basis, then, programmers and organizations are selecting or creating FOSS license terms in an organic process standing in contrast to the OSI's formal evaluation of licenses against the OSD. From the perspective of the Hirschman framework, if the FOSS license is an institutional mechanism synergistically mixing exit and voice, the strength, proportion, and quality of the mix in each particular license may contribute, at some level, to that license's adoption success.

### *C. Evaluating FOSS Licensing Issues in Light of the Framework*

A policy implication of the Hirschman framework is that the beneficial effects of voice to discipline or recuperate a firm are not realized in some instances because there are insufficient institutional mechanisms to effectively allow voice expression. As discussed above, the FOSS license provides a new mechanism with these beneficial effects. It presents a synergistic mix of exit and voice with disciplining effects on proprietary software providers. The exit opportunity offered by the FOSS license provides voice via the threat of exit. FOSS licensed software provides competition springing from a politically different conception of software and its production. This novelty generates the exit opportunity and gives it a unique voice. To the extent the FOSS disciplining force on proprietary software is beneficial, FOSS licensing issues should be evaluated with this beneficial effect in mind.[292] Decision makers, including courts, should assess FOSS licenses in light of the synergistic, reinforcing effects of exit and voice that supply the disciplining force. For close cases of interpretation or doctrine, the FOSS license disciplining effect should be a factor tending toward an outcome that preserves this effect.[293]

---

[292] This approach parallels in a broad way the influence the exit and voice framework has exerted on the issues surrounding corporate governance. *See* Blair, *supra* note 5, at 3-4, 32, 39-43 (arguing that Hirschman's exit and voice framework organizes an evaluative approach for corporate law reform proposals).

[293] The general rubric that preserving FOSS exit and voice synergy should be a factor inclining decision makers in close licensing issues assumes, in the case of judges, that policy considerations should inform judicial decisions, at least at some level. In proposing this rubric, I recognize that the jurisprudential assumption is not without controversy and does not always apply, depending on the nature of the question presented. Moreover, I do not think it necessary to peg the rubric along the continuum

The first potential application of a doctrine that factors in the beneficial disciplining effects of FOSS concerns copyright's doctrine of joint ownership, which, if applied to a FOSS project, would dissolve the web of interdependent copyright permissions that secure the software to FOSS development.[294] The interdependency of the software components assembled into an operable whole combines with the coordination among developers to allow for at least the possibility of joint ownership arguments, depending on the license, its surrounding context, and factual inferences flowing from these.[295] In addressing this possibility, the primary question is whether the FOSS license discourages reasonable claims of joint ownership by a single developer. If not, such a claim would allow a developer to license the software on whatever terms she desired, including proprietary terms.

From the perspective of Hirschman's framework, a successful joint ownership claim has the potential to dilute the voice that rings from the FOSS license and the software in at least two ways. First, the finding of joint ownership would be a noticeable event in the FOSS community, and probably in the greater software ecosystem. A FOSS license "failing" to support the software and its development process would hurt confidence in FOSS licensing, and the particular license at issue would thereafter have a questionable reputation.[296] Second, unlike an allowable fork under a FOSS license, a successful joint ownership claim lets an enterprising joint owner produce a proprietary or dual-licensed version of the software without any attribution to its FOSS origins.[297] This has the potential

---

that runs from rules to standards, because I merely seek to demonstrate that exit and voice synergy is a plausible factor. Its exact invocation and influencing strength will likely vary for judges and policymakers, and vary based on the issue presented. Regardless of its application, it should be counted.

294 If the FOSS licenses underlying the project also grant patent permissions, a finding of joint ownership for copyright in the integrated software does not necessarily imply that joint ownership will exist for the patent rights. Joint ownership arguments for patent rights, especially those which spring from United States doctrine on inventorship, are beyond this Article's scope, other than to note that individually licensed patent rights might still apply even if copyright ownership in the software were found to be joint.

295 *See* NIMMER, *supra* note 9, § 1:4.

296 Among the FOSS licenses cataloged in Part II.A, the corporate-style licenses generally seem to provide the greatest resistance to joint ownership arguments in that their language and structure clearly show the intent to retain individual copyrights and license rights under the FOSS scheme, rather than an intent to jointly own the rights.

297 While attribution-only licenses have not been my focus in this Article, a developer using software licensed under an attribution-only license, such as the BSD-style licenses,

to dilute voice that would otherwise emit from the FOSS version if the proprietary version becomes popular, which it might become if the enterprising developer has distribution advantages in niche markets or other technology to couple with the proprietary version. FOSS software helps carry the movement's message because the licensing approach is novel. If the FOSS software's distribution is lessened because a joint owner successfully distributes a privatized version, its voice-carrying potential is lessened. Under this reasoning, applying the rubric to preserve the synergistic exit and voice effects means inclining against a finding of joint copyright ownership.

The second potential application I will illustrate for the exit and voice framework is known in the FOSS community as the "web services" issue. The FOSS licensing scheme, and in particular the GPL, was put in place before the Internet expanded dramatically in the 1990s.[298] The FOSS license conditions trigger upon a distribution as that term is understood in copyright law.[299] Running FOSS software internally and delivering its functionality to users over the web is generally not thought to be a distribution of the source code, although the answer depends on technological factors as well.[300] An example is the Google search engine. It is reported to run a modified Linux-kernel-based operating system, but it does not make the source code for its modifications available.[301]

Some within the FOSS community are interested in licenses that will create incentives or requirements to apply FOSS licensing to Internet-deployed modifications of FOSS software that companies run on internal computers.[302] One license uses an approach that

---

has a similar opportunity to create a proprietary version, but would still need to retain the attribution notices.

298 *See* MOODY, *supra* note 60, 26-30 (discussing the development of the GNU generally); GPL, *supra* note 7.

299 In this context (and in most other contexts), violation of the distribution right follows from the violation of the reproduction right. *See 2* MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 8.02[A] (2006) (noting that "it is the act of copying that is essential to, and constitutes the very essence of all copyright infringement," including the distribution right).

300 When software runs on a server connected to the Internet, delivering an application to users via web browsers, the source code is not transferred across the network to the remote user. As a result, there is not a distribution of the source code.

301 *See Google's Summer of Code Pays Students to Do Open Source*, *supra* note 89.

302 *See* Free Software Foundation, GPLv3 Rationale Document 1.1, Do No Harm, http://gplv3.fsf.org/rationale (last visited July 24, 2006) (discussing the treatment of

intertwines a technological constraint and a licensing constraint, requiring software released with an ability to view the source code via the web interface to be redistributed with that capability intact.[303] Another license deems distributions to include communicating applications across the Internet as web services.[304] It was anticipated that version three of the GPL would add provisions to handle web services.[305] The January 2006 draft of version three allows those who deploy the license to include web services provisions similar to the first example given in this paragraph, but the main text of the license does not otherwise explicitly provide for web services.[306]

Labeling FOSS that underlies a web service increases indirect voice for the FOSS licensing model. This suggests a minimalist approach for web service, which is to require attribution of the underlying FOSS software. A more expansive approach would create incentives for disclosure of source code modifications. If the mechanism successfully induced source code disclosure, the code would be available to the FOSS development community, with some voice-carrying effect through the FOSS license that induced the disclosure. Moreover, the disclosed code would potentially improve the exit option the software affords, which can translate to a more viable exit threat as voice.

The question in either case is whether a FOSS license, enforced under copyright, or under contract if an assent or agreement is present, will be successful in bringing legal force to the package of incentives that might cause a user or developer to release modifications to FOSS software. There are a number of doctrinal questions here, including the simmering issue of whether FOSS

---

software designed for public use on network servers and differing community views on the matter).

303 Free Software Foundation, *supra* note 44 (discussing the Affero General Public License, which is the GPL with added section 2(d) covering "the distribution of application programs through web services or computer networks"). *See also* Affero Project, Affero General Public License Version 1, ¶ 2(d), http://www.affero.org/oagpl.html (last visited Sept. 19, 2006).

304 Rosenlaw & Einschlag, Open Software License ("OSL") Version 3.0, ¶ 5, http://www.rosenlaw.com/OSL3.0.htm (last visited July 24, 2006).

305 *See, e.g.*, China Martens, *GPL 3 Likely to Appear in Early 2007*, INFOWORLD DAILY, Aug. 4, 2005, *available at* 2005 WLNR 12297689 (noting that the "FSF needs to determine the situation when what's being redistributed is not a copy of the software itself but a service based on that software").

306 GPLv3, *supra* note 7, ¶ 7(d).

software licenses are or should be binding notices (sometimes described as conditional copyright permissions) or full agreements. If display of interfaces via web services is not a distribution of the underlying source code, the distribution right under copyright law might not be available to support the conditional permission approach. Other rights might be available, such as the display right under copyright for audiovisual works embedded in the software interface, but thus far FOSS licenses have paid minimal attention to the display right.[307] Moreover, not all FOSS software will have audiovisual works that qualify for copyright protection. The disclosure condition could be tied to the reproduction or derivative work right, but enforceability is an issue when use is only internal, even though private violations of these rights are actionable.

Basing FOSS licensing conditions on the distribution right helps clarify the enforceability of the baseline FOSS conditions. If a contractual agreement can be obtained, this may enhance enforceability. In any approach, inducing disclosure of modifications underlying web services may be viewed as an attempt to gain greater leverage over FOSS users by FOSS licensing. Some may analogize this to the leverage of the GPL's infectious terms, which have been subject to copyright misuse claims.[308] These new doctrinal questions for web services sit against an existing range of issues for FOSS licensing, most of which find scant guidance in United States case law.

From the perspective of the Hirschman framework, the present goal is not to resolve this sampling of the doctrinal issues raised by web services in FOSS licenses. Like the first application to joint copyright ownership, the point is to preserve the synergistic effects of exit and voice. In the web services context, this means supporting the mechanisms that try to bring legal force to identify or disclose internal FOSS software modifications deployed as web services.

---

307 Besides FOSS licensing paying scant attention to the display right, in general it is perhaps the least familiar of the copyright bundle. R. Anthony Reese, *The Public Display Right: The Copyright Act's Neglected Solution to the Controversy over Ram "Copies,"* 2001 U. ILL. L. REV. 83, 84-85 (arguing that the display right is not familiar, and describing its unique applicability to works, or aspects thereof, transmitted over computer networks).

308 *See* Plaintiff Daniel Wallace's Memorandum on Motion for Summary Judgment, Wallace v. IBM Corp. et al., No. 1:05-cv-678/RLY-VSS, slip op. (S.D. Ind. May 16, 2006), *available at* http://www.groklaw.net/article.php?story= 20050703144738557 (asserting, in an argument embedded in a pro se complainant's antitrust price fixing case, that the GPL is copyright misuse).

The framework should not command the decision, but should be an influencing factor, especially to the extent policy concerns inform the decision.

These two applications of the exit and voice framework for FOSS licensing issues are exploratory. They put aside the full detail of how decision makers should incorporate the framework into licensing and policy decisions. Even with these limitations, the framework's identification of the FOSS license as a unique, institutional mechanism synergistically reinforcing exit and voice garners support for the framework as a viable decision-influencing factor. Beyond this prospective application, the framework suggests a method by which to compare FOSS licenses in the competition that occurs among them in the community and marketplace, and may contribute to the explanations of the tendency for FOSS applications to find success in platform software technology.

## CONCLUSION

As social movements in genesis, FOSS and the labor movement exist apart in time and technology. The labor movement pitted itself against dominant industries of its era. The free software advocates, organized through software, licenses, and the Internet, pit FOSS against influential forces in its era, proprietary software providers. Despite the many obvious differences, the two movements share similarities that include a natural fit with elements of Hirschman's framework in *Exit, Voice, and Loyalty*. That framework has been applied to various issues in employment and labor, and this Article suggests its applicability to FOSS by demonstrating the exit and voice mechanisms in a variety of FOSS contexts. Voice in FOSS licensing corresponds to the political emphasis of the movement. The exit opportunity corresponds to the open source camp, whose emphasis supports economic considerations such as high quality software and bridging the FOSS and proprietary world. This leads to exit alternatives for software users, generating disciplining effects from exit and the threat of exit. FOSS licensing provides a variety of exit opportunities depending on the type of license, with varying degrees of associated direct and indirect voice and unique user participation in the development process. Other forms of voice include extracurricular FOSS contributors and the programs of advocacy, license enforcement, and lobbying that spread the FOSS message, setting up future contributors and adopting users. The Hirschman framework shows that for a complex good, such as

software, both exit and voice can exert disciplining influences on incumbent suppliers. Sometimes these influences are synergistic, and a policy lesson from *Exit, Voice, and Loyalty* is that institutional mechanisms that enable such synergy are to be encouraged.