24

Legal Aspects of Free and Open Source Software

David McGowan[1]

This chapter is about how the law affects free and open source software (F/OSS) development. It discusses the basic principles of copyright and contract law relevant to F/OSS development, and the way these legal principles constitute an environment that sustains the social practices of the F/OSS communities. It also discusses open legal questions and challenges they may present for the future.

Part I discusses the structure of F/OSS licenses—how they are designed to work. Part II discusses some issues and questions regarding this design--whether the licenses actually will work this way if tested. If you are already familiar with copyright and licensing law, you might want to skip Part I and go straight to Part II. Part III discusses two criticisms an influential private firm has leveled at the GNU General Public License (GPL).

---

[1] Associate Professor of Law & Vance K. Opperman Research Scholar, University of Minnesota Law School. My thanks to Dan Burk and Mark Lemley for discussing these subjects with me. Mistakes that remain are my fault. This essay is adapted from David McGowan, Legal Implications of Open Source Software, 2001 UNIV. ILL. L. REV. 241.

Part I

Whether a program qualifies as F/OSS is in one sense a legal question. When developers write code and fix it in a tangible medium, copyright law gives them the exclusive right to reproduce the code, distribute it, and make works derived from their original work. Subject to some important exceptions such as fair use, persons who would like to do these things with code need the author's permission.[2]

Authors grant such permission through licenses. The terms "free" software or "open source" software refer to software distributed under licenses with particular sorts of terms. A common reference guide to such licenses is the "Open Source Definition," which was originally written by Bruce Perens and is now maintained by the Open Source Initiative. It sets out several conditions a license must satisfy if code subject to the license is to qualify as "open source software."[3]

Some aspects of this definition pertain to distribution of code. Programs distributed under a F/OSS license "must include source code, and must allow distribution in source code as well as compiled form." (Posting the source code on the Internet satisfies this requirement.) Such a license "shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources," nor may the license "require a royalty or other fee for such sale." An F/OSS 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." [4]

Much of the attention given to F/OSS development focuses on the GPL's requirement that authors who copy and distribute programs based on GPL'd code (derivative works) must distribute those programs under the GPL. This requirement is specified in Section 2(b) of the GPL, which is the "copyleft" term. I will discuss that term in a moment. In the terminology of the Free Software Foundation (FSF), licenses that require derivative works to be Free Software are "copyleft" licenses. Source-code licenses that allow free copying but do not contain such a requirement are "Free Software" licenses but not "copyleft" licenses.

---

[2] 17 U.S.C. §§201(a); 102; 106.

[3] *See, The Open Source Definition* version 1.9, available at www.opensource.org/osd.html. All references in this chapter to the Open Source Definition are to version 1.9.

[4] Open Source Definition, §§1-3.

The Open Source Definition has some more detailed requirements as well. F/OSS licenses may not discriminate among persons, groups, or fields of endeavor. Other requirements ensure that programmers get credit (or blame) for their work. For example, while a license must allow users to modify the code and make derivative works, it may require them to distribute modified source code in two parts: the original code as written by the licensor and, under a separate name or version number, the licensee's modifications. The definition also states that F/OSS license terms must not extend to other software that is merely distributed alongside code subject to the license. This provision does not pertain to programs that interact with F/OSS code when executed, rather than merely being distributed with them.[5]

Several well-known licenses satisfy the Open Source Definition. I will start with the most famous, the GNU GPL. The GPL sets out a two-pronged strategy designed to enforce the norms of F/OSS development. The first is to have the original author retain the copyright in the author's code or assign it to an entity, such as the FSF, which will enforce these norms. The second is to allow developers to copy, modify, and redistribute the code only so long as they agree to comply with the GPL's terms, which embody at least some of the norms of the F/OSS communities. If a licensee violates the terms, the authors or their assignees may enforce the norms through a copyright infringement action. Courts routinely enjoin the unlicensed use of copyrighted works, so the threat of an infringement action is a powerful enforcement tool.[6]

The GPL helps developers establish and maintain social practices and understandings that perform a nifty bit of legal jujitsu.[7] The GPL employs copyright to suspend the usual operation of copyright within the domain of F/OSS development. This effect of the GPL gets most of the press, and rightly so. Even from a purely legal point of view, however, in its brevity, its clarity, and its creative use of rights, the GPL is an elegant piece of work. Better still, its elegance does not detract from its effectiveness. The GPL is a working document, too. Here's how it is designed to work.

The GPL defines two important terms: "program" means a work subject to the license, and "work based on the program" refers either to the program "or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it either verbatim or with modifications and/or translated

---

[5] *Id.* §§4-6; 9.

[6] On injunctions, *see* 17 U.S.C. §502(a); Cadence Design Sys, Inc. v. Avant! Corp., 125 F.3d 824, 827 n.4 (9th Cir. 1997) (noting that injunctions are presumptive remedy for infringing use); *cert denied* 523 U.S. 1118 (1998).

[7] Professor Benkler was the first to use this metaphor, in his very thoughtful analysis of open source practices. Yochai Benkler, *Coase's Penguin, Or, Linux and the Nature of the Firm*, 112 YALE L.J. 369, 446 (2002).

into another language." The basic license term provides that licensees "may copy and distribute verbatim copies of the Program's source code as you receive it . . . provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty, and give any other recipients of the Program a copy of this License along with the Program."[8]

Licensees may "modify [their] copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work" so long as, among other things, they "cause any work that [they] distribute or publish . . . to be licensed as a whole at no charge to all third parties under the terms of this License."[9] The license also states that these terms apply to "the modified work as a whole" but not to "identifiable sections of that work [that] are not derived from the Program, and can be reasonably considered independent and separate works in themselves" when such independent works are distributed on their own. When independent works are distributed "as part of a whole which is a work based on the Program," however, they are subject to the license as it applies to the whole.[10] Under this model, "each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions."

The GPL further provides that a licensee "may not copy, modify, sublicense, or distribute the Program except as expressly provided under" the GPL. "Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License." In that event, however, "parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties" comply with the GPL's terms.[11] As to how the license binds users in the first place, the GPL says that "by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it."[12]

An illustration may help explain how these terms are designed to work in practice. Imagine three parties: A, B, and C. Suppose A writes a program and distributes it to B under the GPL. A either does or does not give B notice of the

---

[8] GPL ¶0; ¶1.

[9] *Id*. ¶2(b).

[10] *Id*. ¶2.

[11] *Id*. ¶6; ¶4.

[12] *Id*. ¶5.

GPL terms. If he does, then let us assume that B is bound.[13] If A does not give B enough notice to form a binding agreement, then B might argue that she is not bound, but then she has no license—meaning no permission--to copy, modify, or distribute A's code. If B does any of these things without a license, A may sue her for infringement and ask a court to enjoin B's use.

The thing to notice about this part of the structure is that whether B is "bound" by the GPL is really beside the point. Absent a private deal with the author, the GPL is the only thing that gives B the right to copy, modify, or distribute A's code. If the GPL does not apply to B then, if B does any of these things, B infringes A's copyright. It is in B's interest that the GPL apply to B, so there is no logical reason for her to fight it.

Suppose B is bound by the GPL and would like to produce and distribute a work based on A's program. There are two cases to consider here. (I will just identify them now; I discuss them in more detail in Part II.) The first case would arise if B wrote a program in which she copied some of A's code and combined it with some new code of her own to form a single work. Conventional derivative work analysis deals easily with this case.

The second case would arise if B wrote a program consisting entirely of her own code but which interacted with A's code when it was executed. Derivative work analysis is more complex and controversial in this case. A might argue that if executing B's program caused A's program to be copied and to interact with B's program, then the combination of the two programs amounted to a work based on A's program, and therefore to a derivative work under the GPL. (Technically speaking, this claim would be best analyzed as one where the user infringed A's right to make derivative works and B contributed to this infringement by distributing her code.)[14] The FSF has taken this position with regard to at least some programs that interact with GPL'd code.[15] Others disagree.[16] I discuss this disagreement in Part II.

---

[13] *E.g.*, ProCD, Inc. v. Zeidenberg, 86 F.3d 1447 (7th Cir. 1996)(shrinkwrap license); I.Lan, Inc. v. NetScout Serv. Level Corp., 183 F. Supp. 2d 328 (D. MA 2002) (click-through license).

[14] *See* Midway Mfg Co. v. Artic Int'l, Inc., 704 F.2d 1009 (7th Cir.), *cert denied,* 464 U.S. 823 (1983); Sean Hogle, *Unauthorized Derivative Source Code*, 18 No. 5 COMPUTER & INTERNET L. 1, 6 (2001)(discussing contributory infringement argument).

[15] *See* Free Software Foundation, *Frequently Asked Questions About the GNU GPL*, available at http://www.gnu.org/licenses/gpl-faq.html.

[16] *See* Lawrence Rosen, *The Unreasonable Fear of Infection*, available at http://www.rosenlaw.com/html/GPL.PDF .

For simplicity, I will stick with the first case for now. The GPL gives B the right to copy A's code and to modify it to create a derivative work. B's copying and modification of the code is therefore lawful.[17] B therefore owns the rights to her contribution to the derivative work—the original code she wrote herself—and A owns the rights to his code, subject to B's GPL rights.[18]

Suppose B sends her work, containing both A's original code and B's new code, to C. B either does or does not give C enough notice of the GPL terms to bind C. If she does, C is bound by the GPL. If she does not, A might assert that B's failure to give notice to C violated Section One of the GPL. Whether on that ground or some other, suppose B has violated the GPL. Her violation terminates her rights from A. B could no longer copy, modify, or distribute A's code, including any of A's code that B copied into B's derivative work.

If B tried to do any of these things after her GPL rights terminated, A could sue B for both breach of the GPL and for infringement. The most likely result of such a suit would be an injunction preventing B from copying, modifying, or distributing A's code. B would still hold the rights in the code she wrote, which she received by default when she wrote it. As a practical matter, however, this fact might not mean much to B, whose code might be worth little or nothing without A's code.

As to C, if C uses A's code (whether she received it from B or some other source) in a manner inconsistent with the GPL, then A may sue C for infringement. If C adheres to the GPL terms, however, even if she received it from B, whose GPL rights had terminated, then the GPL grants her a continuing right to use A's code.

Section 2(b) of the GPL does not apply to A, who owns all the exclusive rights in the original code. Indeed, some developers run parallel versions of a program, with one version being F/OSS and the other being "private." As discussed more fully in Part II, this fact presents some risk that A might release F/OSS code to the community and then attempt to revoke the GPL rights of his licensees so he could distribute his code solely in binary form for a profit.

---

[17] *Cf* 17 U.S.C. §103(a)(" protection for a work employing preexisting material in which copyright subsists does not extend to any part of the work in which such material has been used unlawfully.").

[18] 17 U.S.C. §103(b)("The copyright in a compilation or derivative work extends only to the material contributed by the author of such work, as distinguished from the preexisting material employed in the work"); Stewart v. Abend, 495 U.S. 207, 223 (1990) ("The aspects of a derivative work added by the derivative author are that author's property, but the element drawn from pre-existing work remains on grant from the owner of the pre-existing work"). The GPL does not vest ownership of the derivative work in the licensor, so a court presumably would consider the author of the new code to hold its rights.

Though A could do this with respect to his own code, he could not take private the contributions of developers who improved that code unless those developers agreed. As noted above, if B had A's permission to write code forming a derivative work, then B owns the rights to the code she wrote. Subsequent termination of her GPL rights to A's code does not change that fact. B is bound by the GPL to release her derivative work under the GPL, so we may presume as a default matter that A receives the derivative program as a licensee under the GPL.[19] If A chooses to incorporate B's code into the original program and take advantage of the improved derivative work then, as to that code, A is a licensee and is bound by Section 2(b).

Under the F/OSS model, programs can easily become (indeed are designed to be) quilts of code from many different authors, each of whom owns rights as to which the others are licensees. As a practical matter, for projects in which more than one developer contributes important work, at least each major contributor would have to agree to "privatize" the code if the project were to be taken private in its most current and complete form. Past a fairly small scale, the web of intersecting and blocking copyrights the GPL creates would make it very hard for any developer to use the code for strategic or anti-competitive purposes.

The GNU project also has created the GNU Lesser General Public License (LGPL), which is designed for certain software libraries "in order to permit linking those libraries into non-free programs." A commercial developer wishing to write programs that interact with GPL'd code might balk at the risk that a court would accept the FSF's interpretation of the GPL and, in at least some cases, treat the developer's program as an infringement of the GPL author's right to make derivative works. Some programs are more valuable if a large number of complementary programs work with them. Some developers therefore might wish to enhance the popularity of their programs by giving commercial firms the option of using F/OSS programs without subjecting the firms' conventionally licensed code to F/OSS treatment. [20]

To achieve this goal, the LGPL distinguishes between programs that contain library material or are derived from library material (a "work based on the library") and those designed to be compiled or linked with the library (a "work that uses the library"). The LGPL provides that works based on a library may be distributed only subject to restrictions similar to those of the GPL. [21]

---

[19]B could release her own code, standing alone, on any terms she chose, so long as standing alone that code did not infringe A's right to make works based on A's program.

[20]LGPL, preamble.

[21]LGPL ¶0 ("work based on a library); ¶5 ("work that uses a library"); ¶2 (restrictions on distribution).

As to a work that uses a library, the LGPL says that, "in isolation," such a work "is not a derivative work of the Library, and therefore falls outside the scope of this License." It also says, however, that "linking a `work that uses the Library' with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a `work that uses the library'." Nevertheless, the LGPL allows a developer to "combine or link a 'work that uses the Library' with the Library to produce a work containing portions of the Library, and distribute that work under terms of [the developer's] choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications." A developer who pursues this course must comply with additional conditions.[22]

As I mentioned earlier, many licenses besides the GPL comply with the Open Source Definition. (As of this writing, the definition lists over 40 compliant licenses.) The other licenses tend to get less press, however, because many of them impose few obligations on licensees, so no one has anything to complain about. The BSD and MIT licenses are examples here. These licenses allow unlimited use of the programs they cover, subject only to obligations to include a copyright notice when distributing the program, a disclaimer of warranties, and for the BSD license, a requirement that the rights-holder's permission be given before the author's name can be used in advertising a work derived from code subject to the license. In this regard, the Apache license is similar to the BSD license.

The important general points are that these are nonexclusive licenses, which means the original author may grant particular users greater rights than are contained in the standard form license, and that without the licenses the legal default rule is that users cannot copy, modify, or distribute code. That means the licenses make users better off than they would be with only the default copyright rule, and the original author has the power to negotiate private transactions that make users better off than they would be with the standard F/OSS license.

Part II

This part discusses the question whether in practice F/OSS licenses will work as designed. I divide the issues between contract questions and intellectual property questions.

---

[22] *Id*. ¶¶5-6.

Contract Law

The key to F/OSS production is the way copyrights are deployed by the various licenses. The legal status of those licenses is therefore an important question, which I explore here using the GPL as an example.

Are F/OSS licenses really contracts?

Contract analysis must begin with the copyright default rule. Absent the author's permission, or a defense such as fair use, users have no legal right to copy, modify, or distribute code. That is true regardless how a user acquires the code; the copyright gives the author rights against the world. That is why, as noted earlier, a downstream user would *want* the GPL to be enforceable, to give the user a defense against an infringement action. At this level, there is really no question whether the GPL works. The code to which it applies is subject to the author's default legal rights, just like any other copyrighted work.

For this reason, the term "contract" fits somewhat awkwardly with F/OSS practices. Contracts involve bargains, which require that the contracting parties exchange something. What do users give authors of GPL'd code? The GPL itself does not seem to obligate users to give the author anything. Indeed, with the exception of the warranty disclaimer, the license does not cover the act of running the code. Only copying, distribution and modification are subject to its conditions. In this sense, the GPL is just a permission to use code.[23] Because it demands no bargain, one could argue that the GPL cannot form a "real" contract.

The difference between granting someone the permission to use code and striking a bargain for its use might seem unimportant, and in most cases it probably is. The difference might be important, however, if a user tried to sue an author on the ground that GPL'd code caused the user harm. A user might try to draw an analogy to real property cases, where property owners owe licensees a duty not to harm them through gross negligence and to warn them of defects in the property, or of dangerous activities on the property, of which the owner is aware but the licensee is not.[24] Or a user might try to rely on laws imposing general duties of care, or to draw an analogy to cases extending duties to anyone who uses property with permission.[25]

---

[23] *Cf Restatement of Property* §512 cmt a ("In a broad sense, the word `license' is used to describe any permitted unusual freedom of action").

[24] *E.g. Restatement of Property* §342.

[25] *See* Cal. Civ. Code §1714(a) (creating general duty of care); Louis v. Louis, 636 N.W. 2d 314 (Minn. 2001) (landowner owes duty of care to all persons invited onto land).

Like F/OSS licenses generally, the GPL ends with a disclaimer of warranties. It makes clear that the author does not vouch for the code and will not pay damages if the code causes harm; users use the code at their own risk.[26] F/OSS code is relatively transparent, and persons who use it are likely well enough informed to fall within the class of persons who know or should know of any dangers in the code, so the risk of liability probably is low. The warranty disclaimer in the GPL might avoid litigation over such matters, however, or at least limit damages in the unlikely event someone files suit.

To the extent a court might otherwise find that authors of GPL'd code owe users a duty, one might find a "bargain" in the users' agreement to relinquish rights in their favor, which a duty might create, in exchange for the rights to use the code. If the requirements of contract formation are met, such disclaimers would work to protect authors against claims for economic harm.[27] There are other situations in which the GPL's status as a contract might be relevant. A small group of developers who actually agreed to share work on a joint project might use the GPL to memorialize their agreement. Or an author might try to terminate the license she granted, and a user might want to use contract or contract-like theories to fight back. In each case, the user would be better off if the court treated the license as a contract. It is therefore worth taking a moment to consider how the GPL fares under traditional contract formation principles.

Can a contract really be formed this way?

So long as authors follow its terms, the GPL fares quite well under conventional contract formation principles. No unusual doctrines or new laws are needed to accommodate it.

The default rule of formation is that a contract may be formed "in any manner sufficient to show agreement, including conduct by the parties which recognizes the existence of such a contract."[28] A licensor may form a contract by giving users up-front notice that their use will be subject to certain license terms, and then allowing users to accept these terms by clicking through dialogue boxes or breaking a shrinkwrap seal.[29] There is no reason why using the software could

---

[26] GPL §§11-12.

[27] *E.g.* Uniform Commercial Code §2-719; Uniform Computer Information Transactions Act §406 ("disclaimer or modification of warranty"); M.A. Mortenson, Inc. v. Timberline Software Co., 140 Wash. 2d 568 (2000); I.Lan, Inc. v. NetScout Service Level Corp., 183 F. Supp. 2d 328 (D. MA 2002).

[28] UCC §2-204(1); Uniform Computer Information Transactions Act §112(a)(2)(assent may be shown by conduct); §202(a) (contract may be formed in any manner sufficient to show agreement).

[29] *E.g.* Forrest v. Verizon Comms, Inc., 805 A.2d 1007 (D.C. App. 2002)(click-through agreement; enforcing forum selection clause) I.Lan, Inc. v. NetScout Service

not count as acceptance of the license terms so long as the user had notice of the terms and the chance to read them before using the code. (Whether particular terms may be enforced in particular cases is, and should be, a separate question.)

The key is to adopt a sensible approach to notice. We do this in physical space all the time. Persons who receive printed forms and do not read them understand that they are agreeing to the substance of the transaction–parking or renting a car, flying on a plane, obtaining a credit card, etc.–plus some contract terms of unknown content. They know that they do not know all the relevant terms, and they willingly proceed on that basis. [30] Persons in such circumstances are protected from opportunism and abuse by doctrines such as unconscionability and unfair surprise.[31]

So long as authors do what it says, the GPL works well within the existing model of contract formation. Section One of the GPL requires licensees to publish "conspicuously and appropriately . . . on each copy" of code they distribute "an appropriate copyright notice and disclaimer of warranty" and to "give any other recipients of the Program a copy of" the GPL. Persons distributing modified code must comply with this term as well and, if the code works interactively, must cause the modified code to display a copyright notice when it is run and tell users how to view a copy of the GPL.[32] If authors comply with these terms, downstream users should be aware of the GPL's conditions when they use GPL'd code.

If everyone is doing what the GPL says they are supposed to do, and the terms of the GPL are placed on distributed code, the formation question resembles a

---

Level Corp., 183 F. Supp. 2d 328 (D. MA 2002)(click-through agreement; enforcing damages limitation); Moore v. Microsoft, 741 N.Y.S. 2d 91 (2002)(click-through agreement enforceable; warranty disclaimer valid); M.A. Mortenson, Inc. v. Timberline Software Co., 140 Wash. 2d 568 (2000)(shrinkwraps); Rinaldi v. Iomega Corp., 41 UCC Rep Serv 2d 1143 (Del. 1999) (enforcing shrinkwrap disclaimer of warranty); Brower v. Gateway 200, Inc., 676 N.Y.S. 2d 569 (1998)(shrinkwrap; enforcing arbitration clause but not choice of arbitrators); Hill v. Gateway 2000, Inc., 105 F.3d 1147 (7th Cir.)(shrinkwrap), *cert denied* 522 U.S. 808 (1997); Micro Star v. FormGen, Inc., 942 F. Supp. 1312 (C.D. Cal. 1996) *aff'd in part, rev'd in part on other grounds* 154 F.3d 1107 (9th Cir. 1998); ProCD, Inc. v. Zeidenberg, 86 F.3d 1447 (7th Cir. 1996)(shrinkwrap license). The leading case criticizing the shrinkwrap method is Step-Saver Data Systems, Inc. v. Wyse Technology, 939 F.2d 91 (3rd Cir. 1991). For a critique of *Step-Saver*, *see* David McGowan, *Recognizing Usages of Trade: A Case Study From Electronic Commerce*, 8 WASH. U. J. LAW & POL'Y 167, 188-193 (2002).

[30] Karl N. Llewellyn, THE COMMON LAW TRADITION: DECIDING APPEALS 370 (1960).

[31] Restatement (Second) Contracts, §211(3).

[32] *Id*. §2(c).

standard form contract situation. The GPL does not require a user to click through a dialogue box, of course, but there is nothing talismanic about that method. It is just one way of making sure that users have notice of license terms and, as importantly, of helping authors demonstrate to a court that they did. The key is to give users notice of the GPL terms in a way that they cannot help but see them, or make a conscious choice to skip over them, before they begin using the code.

A developer who released code with a reference to the GPL and a link to its terms would not comply with the GPL's notice requirement, and would run a greater risk of formation problems. (The link might go dead, for example.) Some developers might follow such an approach, however, and there is still a chance it would be effective as between the original author (who is not bound by the notice requirement of the GPL) and that author's licensees.[33] Though a recent case found that a link at the bottom of a screen did not provide users enough notice that the code they downloaded was subject to a license,[34] when GPL'd code is circulated among developers who are familiar with F/OSS norms and practices, a reference to the GPL combined with a link to a webpage posting its full terms might be sufficiently well understood to justify an inference of assent, even if the full terms of the GPL were not included on each copy of the code.

Does it matter if you don't deal with the author?

A related issue is privity of contract. The basic idea is that only a person who has rights can grant them. If I do not have the author's rights in the original code, but only the GPL's grant of conditional permission to use the code, then how can I give you rights to the code? You can't give what you don't have. Because of this concern, Professor Robert Merges has suggested the GPL may not bind downstream users who take code from someone other than the rights-holder.[35]

---

[33] Persons who redistribute the author's code are bound by the notice provisions, and their failure to place an adequate copyright notice on each copy of the code technically would constitute a breach of their GPL obligations, thus terminating their GPL rights. GPL ¶4. Depending on the circumstance, persons who received code from such breaching parties still might have enough notice to satisfy the formation requirements of contract law.

[34] Specht v. Netscape Comms Corp., 306 F.3d 17 (2d Cir. 2002).

[35] Robert Merges, *The End of Friction? Property Rights and Contract in the Newtonian World of On-Line Commerce*, 12 BERKELEY TECH. L.J. 115, 128-29 (1997) ("what is most significant about the [GPL] is that it purports to restrict subsequent transferees who receive software from a licensee, presumably even if the licensee fails to attach a copy of the agreement. As this new transferee is not in privity with the original copyleft licensor, the stipulation seems unenforceable.").

I do not think this is a significant worry for GPL users, however. The GPL is a nonexclusive, transferable license. It grants licensees the power to distribute code so long as they include the GPL's terms with the distribution. It makes sense to view redistribution of GPL'd code as simply a transfer within the terms of the original license. An analogy might be drawn to a licensee who holds the rights to distribute a movie in North America, who contracts with regional distributors, who may then contract with individual venues to show the work. In addition, one may view authors of derivative works as the licensors of at least their improvements to a program, and perhaps of the derivative work as a whole (though this latter point is less clear). A court holding this view of matters would probably not view a lack of privity as a barrier to an enforceable agreement.

Are the rights irrevocable?

No. The GPL states no term for the rights it grants. Two courts of appeal have held that a license that states no term is terminable according to whatever state law governs the contract.[36] In each case that meant the license was terminable at the licensor's will.[37] Another court, the Ninth Circuit Court of Appeals, which covers the West Coast, concluded that a license that states no term has an implicit term of 35 years.[38] That court based this odd holding on a provision in the Copyright Act that gives authors a five-year window (beginning in the 35th year) to terminate a license agreement regardless of the license terms or state contract law.[39] The court's statutory interpretation was poor, however. Other courts have declined to follow it on this point, and they are right. Outside the Ninth Circuit, it is best to presume that rights holders may terminate the rights of GPL licensees pursuant to applicable state law, which may mean termination at will in many cases.[40]

At least in theory, the ability to terminate at will poses a risk of opportunistic behavior by rights holders. Termination may or may not present a very great

---

[36] Figuring out which state that is would be a significant problem where code is floating around on the Net. I will have to leave that issue for another time.

[37] Korman v. HBC Florida, Inc., 182 F.3d 1291 (11th Cir. 1999); Walthal v. Rusk, 172 F.3d 481 (7th Cir. 1999).

[38] Rano v. Sipa Express, Inc., 987 F.2d 580 (9th Cir. 1993).

[39] 17 U.S.C. §203(a)(3).

[40] The Free Software Foundation's GPL FAQ disagrees with the conclusion I reach here. The FAQ asks rhetorically "can a developer of a program who distributed it under the GPL later license it to another party for exclusive use" and answers " No, because the public already has the right to use the program under the GPL, and this right cannot be withdrawn." http://www.gnu.org/licenses/gpl-faq.html . I am not aware of the basis for this statement.

practical risk, however, depending on the code in question. An initial author could terminate the GPL rights she had granted to use and modify her code, but not the rights licensees had in the code they wrote to form a derivative work.[41]

So, to return to our earlier example, if B wrote a derivative work that used A's code, and if A may terminate the GPL rights he grants, then A may prevent B from distributing A's original code in B's derivative work. Termination presumably would be effective as against persons receiving B's work under the GPL, for B could not give them greater rights to A's code than B had herself. B could, however, continue to distribute her own code, as to which she holds the rights. Whether A's termination was a large blow to the project as a whole would depend on how important his original code was. Whether B's code would be worth anything without A's would depend on the same thing.

Whether A would be likely to terminate would depend at least in part on whether he needed to use B's code, for a termination by A could invite reciprocal termination by B. Projects that incorporate code from many authors, therefore, seem unlikely candidates for unilateral termination. And for projects to which the community has chosen not to contribute its efforts, privatization might do little to disrupt community norms.

So far as I know, the risk of GPL termination is almost completely theoretical. (I discuss in the next section the only case in which it was even slightly tangible.) There is no reason for panic. Future iterations of the GPL and other licenses may well address this question.

Can the rights holder assign the rights? What implications do assignments have?

Authors of F/OSS code may assign their rights. An author might want to assign the rights to an organization willing to police license violations, for example. The organization could then monitor use of the code and take enforcement action where necessary. The FSF serves this role for some projects.[42]

Assignments also might be relevant if developers were sued for distributing code. This issue came up in a suit prompted by a hack of "CyberPatrol," an Internet filter marketed by Microsystems, Inc. [43] In early 2000, Eddy L.O. Jansson,

---

[41] Stewart v. Abend, 495 U.S. 207, 223 (1990).; 17 U.S.C. §103(b).

[42] *See* Declaration of Eben Moglen In Support of Defendant's Motion for Preliminary Injunction on Its Counterclaims, ¶23, Progress Software Corp. v. MySQL AB, No. 01-CV 11031 (D. MA 2002), available at *http://www.gnu.org/press/mysql-affidavit.html*

[43] Microsystems Software v. Scandinavia Online, A.B., 226 F.3d 35 (1st Cir. 2000).

working from Sweden, and Matthew Skala, working from Canada, decided to take Cyber Patrol apart to see how it worked. They were particularly interested in what sites it blocked.

Their efforts produced four things, which they released as package entitled "cp4break." The first was an essay called *The Breaking of Cyber Patrol® 4*.[44] This essay described in some detail the process by which Jansson and Skala were able to discover the encryption and decryption code protecting the filter's database of blocked sites, and how they were able to break the encryption. In addition to the essay, Jansson and Skala released three programs. One of these programs, called cphack.exe, was released in both source and binary code form, and was written to run on Windows. One source file in that program stated: "CPHack v0.1.0 by Eddy L O Jansson / Released under the GPL." Jansson added this message on his own; he meant to tell Skala he had done so, but he forgot.

Microsystems responded to the distribution of the cp4break package with a suit against Jansson and Skala for copyright infringement, breach of contract, and interference with prospective economic advantage. A trial court in Boston issued a temporary restraining order against the defendants.[45] Jansson and Skala did not want to litigate; Microsystems wanted the strongest legal tools possible to prevent distribution of the code. The parties agreed on a settlement with several terms, one of which was that Jansson and Skala assign their rights in the code to Microsystems, which could then (at least in theory) attempt to terminate any rights created by the GPL and proceed on a copyright infringement theory against anyone posting the code.

When news of the settlement broke, media reports questioned whether Jansson and Skala could assign exclusive rights in cphack.exe to Microsystems after having placed a reference to the GPL on the code. Some accounts reported statements that rights transferred by the GPL are irrevocable.[46] As noted above, that is an overstatement. The common law contract rule would allow termination of rights at the will of either party.

Either a prior assignment of exclusive rights or a fixed license term might make it hard for developers like Jansson and Skala to settle such cases. Suppose Jansson and Skala had assigned the rights to cphack.exe to an entity formed to administer GPL rights in the interests of the F/OSS communities. What would have happened then? Microsystems no doubt would have sued the two hackers

---

[44] Copy on file with author.

[45] The restraining order eventually became a stipulated permanent injunction. Microsystems Software v. Scandinavia Online, A.B., 98 F. Supp. 2d 74 (D. Ma. 2000).

[46] *See* Lawrence Lessig, *Battling Censorware*, INDUSTRY STANDARD, April 3, 2000 (quoting Professor Moglen as saying that "GPL is software that cannot be revoked").

anyway. But Jansson and Skala would not have been able to assign the rights in cphack.exe to Microsystems, because they would not have had the rights. Microsystems might have settled for an agreement that Jansson and Skala leave their products alone. If Microsystems really cared about an assignment, however, then Jansson and Skala might not have been able to settle the case as quickly and easily as they did. They would have had to rely on their assignee to assign the rights to the plaintiff to settle the suit.

Similar problems might arise if Jansson and Skala's rights were subject to a fixed term. In that case, their assignment to Microsystems might be subject to that term, which might make the assignment less attractive to Microsystems and make the case harder to settle.[47] For these reasons, the questions of assignment and the ability of authors to terminate GPL rights represent areas of potential tension between the interests of individual authors and the interests of the F/OSS communities.

Intellectual Property Rights

F/OSS production is based on copyright. Even the GPL could not enforce its conditions on copying, modification, and distribution of code without the right to exclude that authors obtain when they fix their work in a tangible medium. Without copyright, there is no copyleft. Even more permissive licenses, such as the BSD or MIT licenses, require the right to exclude to enforce the few conditions those licenses impose.

There is no reason to expect F/OSS development to free itself from copyright.[48] If we assume there will always be opportunistic persons or firms who might try to appropriate a base of F/OSS code for use in a proprietary program, then F/OSS production will always have to rely on the right to exclude being vested in a person or entity willing to wield that right to enforce community norms and thwart appropriation of the community's work. Otherwise developers might find themselves underwriting someone else's profit margins. Developers may do

---

[47] I say "might" here because this is a case in which a court that construed the GPL as simply a permission to use might find that users had no rights to enforce against the author, while a court that construed the GPL as a bargain might reach the opposite conclusion.

[48] Professor Benkler suggests that open source production relies on copyright only to defend itself from copyright, and that "a complete absence of property in the software domain would be at least as congenial to free software development as the condition where property exists, but copyright permits free software projects to use licensing to defend themselves from defection." Benkler, *supra* note 11, at 446. Perhaps open source production would be better off if Congress revoked copyright protection for software; we would have to see. I do not think that is likely to happen, however, so I see no prospect of open source development freeing itself from copyright.

quite a lot of work simply for the joy of it, but their views might change if someone else were free-riding to profit from their labor.

The F/OSS model creates some copyright issues, however. Perhaps the thorniest issue comes up when a developer writes a program that works with GPL'd code but the developer does not want to release that program under the GPL. This question falls under the more general topic of the way the GPL relates to derivative works.

Three portions of the GPL are relevant to the derivative works issue. The GPL defines the phrase "work based on the program," which includes "either the [GPL'd] Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language." Section Two of the GPL states that a licensee "may modify your copy … of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications" if the licensee causes "any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License." In substance, these terms express the view that any program that qualifies under copyright standards as a work derived from a GPL'd work must itself be released under the GPL.

The Copyright Act defines a derivative work as one "based upon one or more preexisting works…." The concept includes some specified categories not relevant here and a catch-all provision encompassing "any . . . form in which a work may be recast, transformed, or adapted."[49] A work "is not derivative unless it has been substantially copied from the prior work."[50] The right to make derivative works therefore overlaps the right to make copies; substantial copying exists at least where an author could maintain an infringement action based on the derivative author's copying.[51] The right to make derivative works is broader, however, because it may be violated even by adaptations or transformations that are not fixed in a tangible medium. The right to copy is not violated unless the copy is fixed.[52]

---

[49] 17 U.S.C. §101.

[50] Litchfield v. Spielberg, 736 F.2d 1352, 1357 (9th Cir. 1984), *cert denied* 470 U.S. 1052 (1985); *See also* H. R. Rep. No. 94-1476 (94th Cong., 2d Sess. (1976))(" to constitute a violation of section 106(2), the infringing work must incorporate a portion of the copyrighted work in some form").

[51] 736 F.2d at 1357.

[52] Lewis Galoob Toys, Inc., v. Nintendo of Am., Inc., 964 F.2d 965, 968 (9th Cir. 1992)(derivative work need not be fixed to infringe author's rights), *cert denied*, 507 U.S. 985 (1993); *See also* H. R. Rep. No. 94-1476 (94th Cong., 2d Sess. (1976))

As noted earlier, there are two ways a program might constitute a work based on a GPL'd program, and thus be treated as a derivative work. The first is uncontroversial. If in writing her program B copies substantially from A's GPL'd program, then B has met the copying requirement and her program will be a work derived from A's program. If B does not have either A's permission to copy A's code or a defense for her copying (such as fair use), then B's production of her program violates A's right to produce derivative works. B may be enjoined from distributing her program, even if she transforms or adapts to new purposes the code she has copied from A.[53]

What if B does not copy A's code but writes a program that, when executed, invokes A's code and combines with it to perform some function? This question is harder than the first, and the answer to it is the subject of some controversy within the F/OSS communities.

The FSF has taken the position that, in at least some cases, a program is subject to the GPL if it combines with GPL'd code when it is executed. The argument is that the combination of the two programs constitutes a derivative work based on the GPL'd program.[54] The quotations in Part I from the LGPL reflect this

---

("reproduction requires fixation in copies or phonorecords, whereas the preparation of a derivative work, such as a ballet, pantomime, or improvised performance, may be an infringement even though nothing is ever fixed in tangible form"). Fixation is not a stringent requirement, however. *See* MAI Sys Corp. v. Peak Computer, Inc., 991 F.2d 511 (9th Cir. 1993)(RAM copies sufficiently fixed to support infringement action); *cert dismissed*, 510 U.S. 1033 (1994). The DMCA partially reversed the holding in this case. 17 U.S.C. §117.

[53] 17 U.S.C. §103(a); Dun & Bradstreet Software Servs, Inc. v. Grace Consulting, Inc., 307 F.3d 197, 210 (3d Cir. 2002).

[54] *See* Free Software Foundation, *Frequently Asked Questions About the GNU GPL*, available at http://www.gnu.org/licenses/gpl-faq.html. Professor Eben Moglen, who also serves as general counsel to the FSF, made the point succinctly in a message posted on Slashdot in February 2003 in response to a developer's question:

> The language or programming paradigm in use doesn't determine the rules of compliance, nor does whether the GPL'd code has been modified. The situation is no different than the one where your code depends on static or dynamic linking of a GPL'd library, say GNU readline. Your code, in order to operate, must be combined with the GPL'd code, forming a new combined work, which under GPL section 2(b) must be distributed under the terms of the GPL and only the GPL. If the author of the other code had chosen to release his JAR under the Lesser GPL, your contribution to the combined work could be released under any license of your

reasoning. In the only reported case in which this issue arose, the court enjoined on trademark grounds the distribution of a program that worked with GPL'd code. The court said the "[a]ffidavits submitted by the parties' experts raise a factual dispute concerning whether the . . . program is a derivative or an independent and separate work under GPL ¶ 2."[55] The case settled without judicial resolution of this issue.

I am sympathetic to the FSF's position on this issue. In simplest terms, the FSF defends the proposition that authors should not have to share the product of their labor with people who will not share with them. The key concept here is the consent of the author (who is free to negotiate a deal under different terms if he chooses), so one could generalize this proposition to say that authors should not be forced to allow others to use their code in cases where authors do not consent to the use.

That proposition in turn could be justified by Locke's theory of property, which holds that persons have property rights in themselves, thus in their labor, and thus in the products of their labor, at least so long as their production does not diminish the quality or quantity of inputs in the common, and thus available to others.[56] One could even add that in *Eldred v. Ashcroft* Justice Ginsburg cited as

---

> choosing, but by releasing under GPL he or she chose to invoke the principle of "share and share alike."

Available at http://interviews.slashdot.org/interviews/03/02/20/1544245.shtml?tid=117&tid=123. The Free Software Foundation has said that when a program employs communication mechanisms normally used to communicate between separate programs, then the modules of code connected are likely to be separate programs under the GPL. It qualifies this conclusion, however, by saying that a different conclusion might be warranted on the facts of particular cases. http://www.gnu.org/licenses/gpl-faq.html.

[55] Progress Software Corp. v. MySQL AB, 195 F. Supp. 2d 328, 329 (D. MA 2002). The court did say in dicta that the GPL licensor "seems to have the better argument here," but concluded that "the matter is one of fair dispute." *Id.*

[56] John Locke, THE SECOND TREATISE OF GOVERNMENT 288 (Peter Laslett Ed. 1988). For discussions of this theory in the context of intellectual property, *see* Wendy J. Gordon, *A Property Right in Self-Expression: Equality and Individualism in the Natural Law of Intellectual Property*, 102 YALE L. J. 1533 (1993); Jeremy Waldron, *From Authors to Copiers: Individual Rights and Social Values in Intellectual Property*, 68 CHI.-KENT L. REV. 841, 849-50 (1993); Justin Hughes, *The Philosophy of Intellectual Property*, 77 Geo. L.J. 287, 288 (1988). Because consumption of information is non-rivalrous, a developer's use of information in writing the commons would not deplete the store of information, thus satisfying Locke's proviso.

one basis for upholding the Copyright Term Extension Act a congressional history of taking into account concerns of "justice" and "equity" for authors.[57]

I suspect many free software advocates would object to this line of argument, however, and it does have its problems. Copyright is more often described as a utilitarian reward system than the embodiment of Lockean theory,[58] and there are utilitarian objections to this approach. There are also significant doctrinal problems. Nevertheless, there is some doctrinal support for the FSF's position, which I will discuss before discussing the problems.

The key to analyzing this question is to identify the original work and the alleged derivative work so the relationship between the two can be examined in relation to the author's rights. *Micro Star v. FormGen, Inc.*, demonstrates the type of analysis needed. That case dealt with the "Duke Nukem" video game. As sold in stores, the game included 29 levels and a utility that allowed players to create their own levels. Players did this by writing files that worked with the Duke Nukem game engine and art library. When executed, player files would instruct the game engine what to retrieve from the art library and how to deploy those images to create the new level. The product of these interactions was a new, player-created, Duke Nukem game level.[59]

FormGen encouraged players to post their levels on the Internet. Micro Star downloaded 300 of these player files, burned them onto a CD, and sold the CD

---

[57] 123 S.Ct. 769, 780 (2003).

[58] Sony Corp. v. Universal City Studios, Inc., 464 U.S. 417, 429 (1984) ("the limited grant is a means by which an important public purpose may be achieved. It is intended to motivate the creative activity of authors and inventors by the provision of a special reward, and to allow the public access to the products of their genius after the limited period of exclusive control has expired"); Fox Film Corp. v. Doyal, 286 U.S. 123, 127 (1932)(government interests in copyright grant "lie in the general benefits derived by the public from the labors of authors.'"); Yochai Benkler, *Siren Songs and Amish Children*: *Information, Autonomy and Law*, 76 N.Y.U. L. Rev. 23, 59 (2001)(" the basic ideological commitment of American intellectual property is actually heavily utilitarian, not Lockean or Hegelian.").

[59] Micro Star v. FormGen, Inc., 942 F. Supp. 1312 (S.D. Cal. 1996), *aff'd in part, rev'd in part on other grounds* 154 F.3d 1107 (9th Cir. 1998). The description of the 29 game levels is at 942 F Supp. at 1314. The build editor referred users to a file containing a license that granted back to FormGen all rights in games created by the players. *Id*. at 1315. Micro Star argued the license was not binding because players were not given adequate notice of its terms before writing their levels. It used this premise to argue that FormGen had waived the right to enforce any rights it might have had in the player levels. The district court avoided this question by finding that Micro Star knew of the restrictions in the license agreement, and that this knowledge was enough to defeat its waiver argument. *Id*. at 1318.

commercially. It then filed suit seeking a judicial declaration that its activities did not infringe FormGen's rights; FormGen counterclaimed for infringement, asking the court to enjoin distribution of Micro Star's CD. FormGen claimed the derivative works at issuer were "the audiovisual displays generated when" its Duke Nukem code was "run in conjunction with" the player-generated files Micro Star distributed.[60]

Micro Star tried to place FormGen between Scylla and Charybdis by advancing two arguments relevant here. An earlier case had held that a work could not be a derivative work unless it was distributed "in a concrete or permanent form." [61] If the derivative work in question was the audiovisual display generated when a player file was executed, Micro Star said, then it did not distribute that work at all, much less in a concrete or permanent form. Micro Star only copied and distributed player files that, when executed in conjunction with FormGen's own code, helped generate the infringing display.

The court rejected this argument on the ground that the infringing audiovisual displays were "in the [player] files themselves," and Micro Star had fixed those files to its CDs. The court understood that the files did not preserve the infringing displays as such, but it thought the displays were "in" the files because "the audiovisual display that appears on the computer monitor when a [player-written] level is played is described--in exact detail—by" a file fixed on Micro Star's CD. The court later said that "[b]ecause the audiovisual displays assume a concrete or permanent form in the … files," the precedent in question "stands as no bar to finding that *they* are derivative works."[62] The italicized language treats the code and the output as one and the same, an approach congenial to the FSF's view.

Having avoided Scylla, however, the opinion was dangerously close to being swallowed by Charybdis, in the form of the rule that a work is not derivative unless it copies from the original. Micro Star's CDs included only the players' code, which FormGen had not written. The player's code invoked FormGen's art library but did not include any material copied from that library. Micro Star pointed out that "[a] work will be considered a derivative work only if it would be considered an infringing work if the material which it has derived from a prior work had been taken without the consent of a copyright proprietor of such prior work," and argued that because the player files did not copy FormGen's code they could not be derivative works.[63]

---

[60] 154 F.3d at 1109-1110.

[61] Lewis Galoob Toys, Inc., v. Nintendo of Am., Inc., 964 F.2d 965, 967 (9th Cir. 1992), *cert denied*, 507 U.S. 985 (1993).

[62] 154 F.3d at 1111-12 (emphasis added).

[63] *Id*. (quoting United States v. Taxe, 540 F.2d 961, 965 n.2 (9th Cir. 1976).

The court rejected this argument on the ground that the original work at issue was the Duke Nukem "story," which Micro Star infringed by distributing code that, when executed with FormGen's code, generated what were in effect sequels to that story. The court noted that the player-written files at issue would only work with FormGen's code, and said in passing that if these files could be used by some other program to generate some other story, there would in that case be no infringement of FormGen's rights.[64]

This qualification suggests that the opinion is best understood as applying a contributory infringement theory of liability, though the court did not decide the case on that ground. If it is the display that infringes, then the infringer is the person who creates the display. In *Micro Star*, that would be the player who runs Micro Star's player-developed files in conjunction with the Duke Nukem game. Micro Star might be liable for contributing to this infringement by distributing the files, but then its liability would depend on whether the files had substantial non-infringing uses.[65] Because the player files could not work with any other program, that issue would have been decided in FormGen's favor, meaning the court reached a sensible result on the facts before it, even if one might debate its doctrinal analysis.

Micro Star offers some support for treating works that interact with GPL'd code as derivative works subject to the GPL. The court did find that a program not copied from an author's code could be enjoined as infringing the author's right to create derivative works because the program produced an infringing audiovisual display when executed in conjunction with the author's code. To that extent, it supports the proposition that a work may infringe the right to create derivative works by interacting with an existing program. In addition, the earlier case that held a derivative work must assume a concrete or relatively permanent form, a rule that presents a problem for the idea that interoperation creates a derivative work, was probably wrong on that point.[66] *Micro Star* undercut the earlier holding, thus strengthening the case for the FSF's position.

If one takes seriously the notion that the derivative work at issue was the audiovisual display of a "sequel" to the Duke Nukem "story," however, then the point of the case is the output and its relation to a protected story, not the interaction of code. On this reading, the case does not imply anything about the interaction of code that does not produce infringing output.

One could of course try to extend this holding to cases where code interacted but did not produce infringing output. There is some authority for that extension.

---

[64] *Id.* at n.5.

[65] *See* Sony Corp. v. Universal City Studios, Inc., 464 U.S. 417, 442 (1984).

[66] *Galoob*, 964 F.2d at 967.

In *Dun & Bradstreet Software Services, Inc. v. Grace Consulting, Inc.*,[67] the Third Circuit found infringement as a matter of law where a consultant both copied and modified code and wrote programs that invoked the rights-holders' code when executed. [68] Because the case involved literal copying and license violations as well as the writing of programs that interacted with the plaintiff's code, and because the defendant's programs appeared to function as substitutes for upgrades to the plaintiff's original programs, rather than as complements, it is hard to determine the opinion's reach on the derivative works issue. Nevertheless, in *Dun & Bradstreet* there was no infringing output similar to the audiovisual display at issue in *Micro Star* (the output most clearly at issue was a W-2), so it fits better with the FSF's position than does *Micro Star*.

Notwithstanding this authority, there are problems with the proposition that one creates a work derivative of a program just by writing another program that interacts with it. At the simplest level, neither the statutory language nor the language of the GPL supports the argument very well. It is a stretch to say that a program that interacts with GPL'd code "recast[s], transform[s], or adapt[s]" that code, as the statutory language requires.[69] It is more natural to say the

---

[67] 307 F.3d 197 (3d Cir. 2002). As of this writing, Grace Consulting had pending before the Supreme Court a petition for a writ of certiorari to the Third Circuit.

[68] The case involved a successor to Dun & Bradstreet Software Service, called Geac, and a consulting firm called Grace. At one point, citing *Micro Star*, the court said "[u]nless authorized by Geac, its right to create derivative works has been usurped by Grace, whose product instructs the computer to incorporate Geac copyrighted material with its W-2 program." *Id.* at 210. The court later said "Grace's W-2 program using Copy and Call commands copies Geac's computer copyrighted code. Thus, it is a derivative work; the inclusion of the Copy and Call commands makes Grace's W-2 programs infringing, derivative works of Geac's copyrighted software." *Id.* at 212. The court later rejected Grace's arguments (i) that its "Copy command does not modify the code"; (ii) that "industry practice uses the commands `to interoperate two systems;'" and (iii) that "the Copy command does not insert text from one program into another; their program remains separate in memory." *Id.* at 213. The court said "Grace admitted that the installation, testing, compiling and link editing of its W-2 programs required copying Geac's software and link editing the Geac code. Geac therefore argues that these trial admissions compel the conclusion that, `as a matter of Law,' Grace's W-2 programs are infringing because they contain copies of Geac's copyright code and are derivative works of Millennium. We agree." *Id.* These statements are the strongest support of which I am aware for the FSF's position on derivative works.

[69] *See* 17 U.S.C. §101; Ty, Inc. v. Publications Int'l, Ltd., 292 F.3d 512, 520 (7th Cir. 2002) ("A derivative work thus must either be in one of the forms named or be `recast, transformed, or adapted.'"); Castle Rock Ent., Inc., v. Carol Pub. Group, Inc., 150 F.3d 132, 143 (2d Cir. 1998) ("derivative works that are subject to the author's copyright transform an original work into a new mode of presentation").

program simply runs the code, causing it to do no more than it was designed to do in the way it was designed to do it. And, as the general counsel of the Open Source Initiative has pointed out, the GPL does not refer to "combining" one work with another.[70] The definition of a "work based on a program" piggybacks on the legal definition of derivative works, and the copyleft provision itself refers to a work that "contains or is derived from the Program."[71]

Piggybacking on that legal definition creates problems because programs that work with GPL'd code but do not copy from it do not, standing alone, satisfy the requirement that a derivative work copy from the original. That means the programs are not in and of themselves derivative works, which is indeed the position taken in the LGPL. But if it is only the combination of the programs that is the infringing work, then the person who combines them is the infringer. On the FSF's account it is the individual user who infringes; the author of the program that works with GPL'd code is at worst a contributory infringer.

Because the derivative works argument in this context is a contributory infringement argument, it is subject to two defenses. Both these defenses rest on the facts of particular cases, so they cut against general statements that invoking GPL'd code creates a derivative work. They do not mean interoperation cannot create a derivative work, but they call into question the proposition that it always does.

First, if users who create the combined work by executing the programs have a defense, such as fair use, then there is no infringement. In that case, the author of the program that works with GPL'd code would not be liable; one cannot be liable for contributing to something that did not happen.[72]

Second, the author would not be liable for contributory infringement if the program in question had substantial non-infringing uses. For example, suppose the program at issue combines both with GPL'd code written by an author who claims the combination is a derivative work and with other programs not subject to the GPL, or with other GPL'd programs whose authors do not object to interoperation. Assume these facts mean that, in those applications, the program does not infringe anything. In that case, the program would not be subject to liability for contributory infringement on the ground that some

---

[70] Lawrence Rosen, *The Unreasonable Fear of Infection*, available at http://www.rosenlaw.com/html/GPL.PDF.

[71] GPL ¶2(b).

[72] *See* Lewis Galoob Toys, Inc., v. Nintendo of Am., Inc., 964 F.2d 965, 970 (9th Cir. 1992), *cert denied*, 507 U.S. 985 (1993) (discussing possible fair use defense by users of utility that modified output from game console; rejecting claim that defendant could be liable for contributory infringement even if consumers did not infringe); 3 Melville B. Nimmer and David Nimmer, NIMMER ON COPYRIGHT §12.04(3)(a)(2003).

infringing uses should not stifle development and distribution of devices that do more than break the law.[73]

Perhaps more fundamentally, the economic justification for the derivative right weakens when it is extended to a program that does no more than cause another program to run as it was intended to run. Actual transformations or adaptations of a program satisfy demand in different ways than does the original. As the *Micro* Star court's analogy to movie sequels pointed out, players who purchased the original 29 levels of Duke Nukem got added value from the additional levels written by other players, which amounted to "new" stories. A utility that sped up or slowed down play by altering data generated by the game might add that kind of value, too.

In those cases, the derivative right allows the author to capture revenues from the value added by transforming their original work. Authors could not capture that revenue in their original sale because they did not sell the transformed work. When a program is not adapted to do anything other than what it is originally distributed to do, however, there is no adaptation or transformation value added for the original author to capture. The author presumably charged in the original sale the profit-maximizing price for ordinary uses of his program, so there is at best a weak case for treating as derivative works programs that do no more than invoke the ordinary operations of his original program. [74]

Against this point, one might say such concerns are irrelevant to F/OSS development. Unlike conventional commercial development, F/OSS development is not about capturing all the monetary value of one's work, but about the principle of share and share alike. This difference is real and it is important. It is a difference in the way different developers use copyright law, however. It is not a difference embodied in the law itself. For a court to treat programs that combine with a GPL'd program as derivative works of that

---

[73] Sony Corp. v. Universal City Studios, Inc., 464 U.S. 417, 442 (1984). It may seem odd that authors of *other* GPL'd programs could frustrate the claims of an author who objected to interaction, but the contributory infringement standard compels this result. *Sony* provides an example of this point. In discussing why the video cassette recorders at issue in that case had substantial non-infringing uses, the court pointed out that many rights-holders, such as the producers of public television programs (Mr. Rogers) or the National Football League, were happy to have their works recorded. 464 U.S. at 445-46. As the Court put it, "in an action for *contributory* infringement against the seller of copying equipment, the copyright holder may not prevail unless the relief that he seeks affects only his programs, or unless he speaks for virtually all copyright holders with an interest in the outcome." *Id*. at 446.

[74] *Cf* Lee v. A.R.T. Co., 125 F.3d 580 (7th Cir. 1997)(Easterbrook, J.)("Because the artist could capture the value of her art's contribution to the finished product as part of the price for the original transaction, the economic rationale for protecting an adaptation as `derivative' is absent.").

program, the court would either have to extend the derivative works concept as a whole beyond the economic rationale that justifies it in ordinary cases or create a special rule for F/OSS programs. The Copyright Act provides one definition for all derivative works, however. Neither that definition nor the cases interpreting it supply a premise that might justify such a distinction.

A significant strength of the GPL with regard to other issues is that it requires no such special treatment. It implements the principle of "share and share alike" using doctrines and arguments no conventional developer could protest. In such cases, developers using the GPL can do quite well asking judges to do just what they normally do in copyright cases. That would not be true if the developer had to ask the judge—a busy generalist who probably is not familiar with software development—to distinguish F/OSS cases from other cases.

Lastly, there is a utilitarian concern. A rule that one program may form a derivative work of another by interacting with it would make it harder for developers to write interoperable programs. Cases have recognized a fair use defense to infringement where transformative users copied code in the process of reverse engineering to gain access to uncopyrightable interfaces, to which they then wrote programs. The defense extends to copying needed to test such programs to make sure they would work with hardware such as a game console.[75]

The copying of copyrighted material at issue in these cases was ancillary to the development of programs that worked with unprotected interfaces, so the letter of these holdings does not extend to routine copying of protected code, which is the problem raised by a program that interacts with another program. Still, the leading case said the increase in computer games compatible with a particular console--an increase attributable to reverse engineering and the copying it entailed--was a "public benefit." The court also said "it is precisely this growth in creative expression, based on the dissemination of other creative works and the unprotected ideas contained in those works, that the Copyright Act was intended to promote."[76] It is possible that these cases undervalue competition between platforms and *de facto* standards and overvalue competition within them.[77] Regardless whether that is true, however, the concerns these cases express are legitimate and, in any event, are reflected in current copyright doctrine.

---

[75] *E.g.* Sony Computer Entertainment, Inc. v. Connextix Corp., 203 F.3d 596 (9th Cir.), *cert denied* 531 U.S. 871 (2000); Sega Enters.s, Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992).

[76] *Sega*, 977 F.2d at 1523.

[77] *See* Joseph Farrell & Michael L. Katz, *The Effects of Antitrust and Intellectual Property Law on Compatibility and Innovation*, 43 ANTITRUST BULL. 609 (1998); David McGowan, *Innovation, Uncertainty, and Stability in Antitrust Law*, 16 BERKELEY TECH. L. J. 729, 807-08 (2001).

The position that works that combine with GPL'd code are derivative works of that code is in tension with the values these courts cited in holding that reverse engineering to achieve interoperability is a fair use of copyrighted works. It is true that developers who write programs that work with GPL'd code are free to release their programs under the GPL, thus eliminating the concern, but it is often true that an independent developer could achieve interoperability by taking a license. From the perspective of copyright policy, the question is how far the author's rights should extend into the space surrounding their work.

No court has actually decided this question, so it would be a mistake to suggest there is a clear-cut answer. Nevertheless, current doctrine and utilitarian considerations suggest that courts are not likely to extend the derivative works concept to programs that interact with a GPL'd program but which are neither created by copying code from it nor transform it into something analogous to a sequel. There may be unusual cases in which it makes sense to treat interoperation as the creation of a derivative work. Whether there are and what they might be will have to be worked out by judges in the context of the facts of particular cases.

Part III

Debates over F/OSS development practices and licenses have become common as the GNU/Linux operating system has become more popular, and conventional firms such as Microsoft have come to see the F/OSS communities as more of a threat to their business models. Much of this debate concerns general matters of social ethics or economics, which are addressed elsewhere in this volume. Some of it concerns legal issues, however, a couple of which warrant brief comment here. The main point is that the most prominent legal criticisms of the GPL are actually criticisms of copyright; they do not establish any difference between code distributed under conventional licenses and code distributed under the GPL.

The first criticism is that the GPL is a "viral" license that might "infect" proprietary programs. Microsoft at one time posted an FAQ that said the GPL "attempts to subject independently-created code (and associated intellectual property) to the terms of the GPL if it is used in certain ways together with GPL code." On this view, "a business that combines and distributes GPL code with its own proprietary code may be obligated to share with the rest of the world valuable intellectual property (including patent) rights in *both* code bases on a royalty free basis." Variations on this theme run throughout the FAQ and statements by Microsoft executives. [78]

---

[78] Some Questions Every Business Should Ask About the GNU General Public License (GPL), question 3 ("How does your use of GPL software affect your intellectual

There is no reason to believe the GPL could somehow force into the F/OSS worlds code a private firm wanted to treat as a commercial product. At worst, whoever held the rights to a GPL'd program could try to enjoin the firm from distributing commercially a program that combined with the GPL'd code to form a derivative work, and to recover damages for infringement.[79] In cases where a program actually copied code from a GPL'd program, such a suit would be a perfectly ordinary assertion of copyright, which most private firms would defend if the shoe were on the other foot. A commercial firm producing a program that created a derivative work because it copied a GPL'd program could avoid such litigation by writing its own code instead of copying someone else's.

The criticism is really directed at the second type of derivative work argument, based on interoperation, which I discussed in Part II. If courts adopted a special definition of derivative works that applied only to F/OSS code, then the "viral" criticism might make a legitimate point against the GPL that would not apply equally to conventionally licensed code. There is little if any chance that courts will do that, however. If courts reject the general idea that interoperation alone creates a derivative work, the "viral" criticism is moot. If courts adopt that general view, then the resulting doctrine of derivative works would apply equally to both F/OSS and conventionally licensed code.

Even in that case, however, so long as a firm did not infringe any rights when it was writing a program that created a derivative work when executed (as opposed to a program that constituted a derivative work because it contained copied code), the firm would still hold the rights in its work. The program itself would not be a derivative work; at worst it would be a tool that contributed to the infringing creation of a derivative work by the user who executed the program in conjunction with GPL'd code.

If the program at issue worked with programs other than the GPL'd program written by our hypothetical plaintiff, then the firm might be able to establish that the program had substantial non-infringing uses, which would defeat the contributory infringement claim.[80] Even if the firm were found liable for contributory infringement, however, in this type of case there is no reason to believe such a finding would deprive the firm of its rights in its work, rather than subjecting it to an injunction against distribution and to damages.[81]

---

property rights"). As of this writing, Microsoft seems to have taken down the FAQ. A copy is on file with the author.

[79] 17 U.S.C. §504.

[80] *See* Sony Corp. v. Universal City Studios, Inc., 464 U.S. 417, 442 (1984).

[81] I am not aware of a case that deals with this question. Section 103(a) of the Copyright Act provides some support for the position taken in the text, however. It states that protection for works "employing preexisting material in which copyright subsists does not extend to any part of the work in which such material has been used

Microsoft's FAQ also suggests that the GPL's disclaimer of warranties leaves users vulnerable in the event a GPL distribution infringes a third party's rights. This point is partly sound.[82] The true author would not be bound by the GPL, which would therefore provide users no defense if she asserted her rights. And it is possible that the disclaimers in the GPL might prevent users from seeking recourse against whoever gave them the code under the GPL. Commercial vendors commonly disclaim warranties, too, however, including warranties of non-infringement.[83] This risk is therefore properly attributable to copyright licensing practices generally, rather than to the GPL in particular.

Conclusion

As a legal matter, F/OSS production confirms the wonderful versatility of a system that creates general property rights and allows individuals to deploy them in the ways that best suit their needs. F/OSS production rests ultimately on the right to exclude, but the point of the right in F/OSS communities is that it is not used. Like the sword of Damocles, the point is not that it falls, but that it hangs.

The main point of F/OSS licenses and practices is the social structure they support–the opportunities they create, the practices they enable, and the practices they forbid. The achievements of the F/OSS communities are, of course, mostly a testament to the community members who have taken advantage of the opportunities these licenses have created. But the licenses are an elegant use of legal rules in a social context, and should be appreciated on those terms.

---

unlawfully." 17 U.S.C. §103(a). If the derivative work at issue is the *combination* of a firm's program and GPL'd code, then Section 103(a) would deny the firm rights *in that combination*. In the hypothetical case at issue here, the original program would not itself employ preexisting GPL'd material, so Section 103 would not deny it copyright protection. (My thanks to Mark Lemley for this suggestion.)

[82]So far as I know, however, there is no evidence to support the FAQ's ungenerous implication that open source developers will knowingly distribute infringing code, counting on the GPL to protect them. *See Some Questions Every Business Should Ask About the GNU General Public License (GPL)* §11 (" You should also ask yourself if GPL developers may conclude that this disclaimer makes it okay to distribute code under the GPL when they *know* they don't have the rights required to do so").

[83] *See* Uniform Computer Information Transactions Act §401(d)(allowing disclaimer of warranty of non-infringement).