Basic open source software licensing

Information to get started understanding open source licensing.

What is source code?

Source code is the text file that a computer programmer uses to write a program. Source code files are processed through a program called a “compiler,” which creates object code or binaries — the machine language files that actually run on your computer. Most programming languages are written this way, including FORTRAN, C, C++ and Java. Some programs are not executed in object code; they are executed by interpreters or in markup languages where the source code is essentially executed as is. These include HTML, PERL, and most BASIC.

What is Open Source Licensing?

Here is a definition of open source.

“Open Source” is sometimes also called “free software,” but as the FSF says, think free speech, not free beer. Open source licensing means licensing of software with the source code available to licensees, so the licensees can make changes to the software. It doesn’t mean all software should be free of charge. Back in the 1970’s, most software programs were licensed in source code form, because most software was developed custom for the client. In the late 1980’s, when PCs began to be standardized, programs became standardized, too. So developers began developing non-custom, off-the shelf software. They also began distributing object code only,
and keeping the source code from the user. Open source advocates want to reverse this practice, so all users have access to source code.

Why don’t licensors license source code?

Most software developers only license the object code to their programs for two reasons: (1) they do not want competitors to see how to decode their programs, and (2) they do not want licensees to modify the programs, because they are concerned about technical support problems arising from bugs introduced by “do-it-yourselfers.” On the other hand, open source advocates don’t think it’s fair to have to rely on the licensor to make all fixes and changes.

OK, so why does any developer license source code?

Some licensees have the bargaining power to demand it.

Some licensors are required to use open source licensing because they have used code they got under the GPL.

Some licensors believe free software is in the best interests of everyone.

What is GNU? Linux? The GPL? The FSF?

There is plenty of information on the Web about this. Start with www.fsf.org, the site for the Free Software Foundation. Linux is an open-source alternative to Unix — the operating system used by many large computer systems. Linux is now used as alternative to many other systems, such as Windows and embedded systems. GNU is a recursive acronym for “GNU’s Not Unix,” the name of the project under which Linux is developed and promoted. The Linux kernel is licensed under the GNU General Public License or GPL.

Are there other open source agreements beside the GPL?
Plenty! Check out Mozilla (my personal favorite), BSD, and Apache.

I don’t have to worry about the GPL do I? I’ve heard it can’t be enforced in court.

*Enforceability.* Most unenforceability arguments are based on the fact that the user never accepts the terms of the GPL. In other words, most people have made an argument based on lack of formation. However, “shrink wrap” and similar unsigned software license agreements are considered enforceable today. So this argument is probably not worth relying on.

*Willingness to Enforce.* Most GPL code (such as the Linux kernel) is owned or licensed by the FSF. The FSF has publicly expressed its willingness to enforce the GPL, and has an active informal enforcement program in place.

*What about MySQL? Did that prove the GPL is enforceable?* The only court case involving the GPL so far (MySQL v. NuSphere) did not involve FSF-owned code. Also, it was primarily a trademark dispute, and was decided on grounds other than the enforceability of the GPL. In any case, the NuSphere product at issue (Gemini) was statically linked to GPL code. This is a far more clear-cut case than most software companies face when making GPL compliance decisions, so a decision in that case would have left many open questions.

*Business Consequences.* For the GPL, the business consequences of engendering the ill will of the development community can be worse than any lawsuit.

*What happens if the FSF sues me? Will I have to lay open my source code?* This question can’t be answered briefly. If you want help assessing the consequences of violating the GPL, you should consult with an attorney.

Can you help my with my open source licensing questions?

*Clients.* If you are looking for advice particular to your situation, you may wish to seek my advice as an attorney. I have lots of experience in consulting on open source questions, and I am reasonably conversant in C++ and other languages. If you are an attorney in a more general practice, you may wish to refer clients to me to get expert advice in this area. You may find this...
to be an attractive alternative after you read the GPL — a very difficult agreement to analyze and interpret.

There are lots of software lawyers. Why should I call on you for my open source licensing questions?

If your software lawyer does not get this joke: `#define ever (;)` s/he may not know how to analyze, say, the complexity of data sharing of inherited classes to determine whether two dynamically linked C functions must both be licensed under the GPL. (Wish I could take credit for the joke but I can’t. It was told to me by a teacher in a programming class.) Also, many lawyers use a traditional approach GPL to compliance analysis — which misses the point entirely. They read the agreement and carefully analyze its language. Usually, this is the right way to interpret a legal agreement, but the GPL is not written in traditional legal style, and analyzing the GPL is not just a legal task — it requires knowledge of programming, and knowledge of the industry.

What is open source science/research/biotech?

It’s not very precise to apply the term open source in areas other than software. Open source is about access to source code — information that enables full use of the software. Most other contexts don’t have an exact analogy. When people use terms like this, they usually either mean community-developed works or open (i.e. patent-free or RAND licensed) standards. There are even people who talk about open source yoga and religious texts! The idea of openness is a good one, but open source is probably best left to describe software.
What are the most difficult questions in open source licensing?

*These are the things my clients and I think about a lot.*

What is the scope of derivative works under the GPLv2?

I have often called this the $64,000 question of open source licensing. (Some of you are too young for that reference, but it comes from a game show in the 1960’s — back then, that much bought you a house, not just a Mercedes Benz E Class.) There are some relatively easy cases to determine GPL scope, and you will find more about those on the FSF’s FAQ about GPLv2 on my links page. But there are other cases, such as those involving Linux kernel drivers, dual processors, virtualization layers, that are more complicated. To analyze these questions, one needs some technical understanding about how software is put together, and also an understanding of the legal principles underlying GPL.

What constitutes distribution under GPL and other copyleft licenses?

It’s pretty well understood that the copyleft requirements of most licenses only adhere when one actually distributes copies of code. (That isn’t true for all the copyleft licenses, just the most common ones.) But distribution can become a complicated question, particularly when copies are provided to developers, testers, corporate affiliates, and outsource providers. This is a situation where the right legal help can be of great benefit in managing the need to comply with copyleft.
Are there implied patent licenses under GPL and other open source licenses?

Although some open source licenses have express patent licenses, none of them have reservations of rights. So, there may be “implied licenses” granted when one contributes to or distributes open source code. This causes great concern for companies that have patent portfolios to protect. The hard part is not usually patents that would read directly on the code — it’s of greater concern how far beyond that an implied license would go. The law on implied licenses is very unclear, and overlaps with notions of patent exhaustion and estoppel.

How do I comply with notice requirements for embedded software?

Notice requirements in open source licenses aren’t usually complicated, but they can be an enormous headache. The more complex the product, and the less it looks like application software, the more difficult it is to apply notices consistently and rationally and still be compliant. Companies use a lot of different practices, most of which are a compromise of some sort. Coming up with a notice procedure takes knowledge of open source requirements, but also common sense and a practical approach.

Does open source licensing allow or require me to use trademarks?

Open source licenses are not trademark licenses. But there is a tension between trademark law and open source licensing. Trademark law covers when you can, or must, use the trademark, logo or brand name of a company that is a source for software. It also covers when you must not. Most companies that release open source software should carefully consider what brands or identifiers they will apply to that software, and enunciating transparent policies for trademark use.

SHARE THIS:

Like

Be the first to like this.
What should my company know about open source?

A "cheat sheet" for entrepreneurs, lawyers, and engineers.

Do we need an open source policy?

You have a policy, whether it is written down or not. It could range from "no open source at all" to "anything goes." The question is: does anyone follow it? Is it sensible for your business? Written policies are useful to communicate your expectations about use of open source in your organization, particularly to outsourced developers or engineers in subsidiaries and affiliates, whom you may not see every day.

My company doesn't use open source. Why should I care about it?

You are wrong. Your company uses open source. You just don't know it yet. See question #1 about your policy.

Should my engineers contribute to open source projects on their own time?

It's likely they are already doing so, but they may need help understanding whether they are encumbering your intellectual property rights or complying with your expectations. Some companies don't allow their engineers to contribute, but there can be a downside to that. The top

https://heathermecker.com/open-source-faq/what-should-my-company-know-about-open-s...
engineers will want to contribute, and by contributing, you may be able to set the direction of open source projects so they will meet your needs.

I heard the GPL isn’t enforceable. Is it OK to violate it?

That’s wrong, and no.

I read the GPL and don’t understand it. Help!

That’s OK, most people don’t. You can’t understand it very well by reading it on its face. (Otherwise there would not be many long FAQs about it on the FSF web site.) Most questions about the meaning of GPL have straightforward answers, but if you are asking, for instance, what is a “derivative work,” or what is “distribution,” you probably need expert help.

Using open source seems complicated? Is it worth it?

Usually it is, but there is no one answer. Open source licensing works better for some software than other software. The market seems to think it worthwhile — most companies use it.
Open Source Community Over-REACTs to X Rated Code

Recently, Apache re-classified code under Facebook’s “BSD+ Patents” license to “Category X,” effectively banning it from future contributions to Apache Foundation projects. The move has re-ignited controversy over the patent grant, but like many events in the open source community, the controversy is more partisan than practical. In fact, it’s unlikely the move will affect adoption of ReactJS, and the criticisms of the BSD+patent grant mostly don’t survive the scrutiny of reason.

The Facebook patent grant, officially called the Additional Grant of Patent Rights Version 2, has been in effect for years. It applies to the wildly popular ReactJS code — a Javascript library for rendering user interfaces. The roster of major technology companies using the code is impressive, including such consumer-facing giants as Netflix — and of course, Facebook itself.

A New Reaction to an Old Grant

The reaction to this news is surprising, given the parallel patent licensing model is nothing new. Facebook released its “BSD+Patents” grant in 2013 (with a revision in 2015). But a similar model was used with some fanfare by Google with its WebM codec in 2010. This licensing model involves two parallel and simultaneous grants of rights: a BSD license to the copyright in the software, and a separate grant to practice patents that read on the software. Putting the two together means there are two independent and parallel grants of rights. In this respect, it is quite similar to the Apache 2.0 license which, like BSD, is a permissive license, and which also contains a defensive termination provision that exists alongside the copyright license grant.
Much of the reaction to Apache Foundation’s announcement has just created confusion, such as this article misleadingly calling it “booby-trapped.” In fact, many open source licenses have defensive termination provisions — which are mostly considered a reasonable mechanism to discourage patent lawsuits, rather than a booby trap. They are also the rule rather than the exception; all major open source licenses with patent grants also have defensive termination provisions — each with slightly different terms. The difference between the Facebook grant, which Apache has rejected, and the Apache 2.0 license, which Apache requires for its projects, is more subtle than the controversy suggests.

Defensive Termination Provisions Come in Many Flavors

Defensive termination provisions vary in two main ways: the trigger for termination, and the scope of rights terminated. As to the scope of rights terminated, there are two camps: those that terminate only the patent rights grant (including Apache 2.0, Eclipse Public License, and the Facebook grant) and those that also terminate the copyright license as well (Mozilla Public License and GPL 3). In other words, for most licenses, bringing a patent infringement suit can only cause termination of one’s patent rights; for the others, bringing a patent lawsuit can result in termination of the copyright license as well — forcing one to stop using the code. Copyright license termination is a much stronger anti-patent mechanism, and more risky for private businesses, resulting in some private companies refusing to use GPL3 or MPL code.

The Facebook grant differs from most other open source licenses in its threshold for triggering termination. In Apache 2.0, for example, the termination of the patent grant is triggered by a patent claim accusing the software provided under the license. The idea is to create a “patent commons” for the software. Most other open source licenses follow roughly this calculus. The Facebook patent license also terminates if the licensee brings a claim against Facebook, or against any party accusing a Facebook product. In that respect, the termination trigger is similar to the one in the Common Public License 1.0, written many years ago by IBM. (“If Recipient institutes patent litigation against a Contributor with respect to a patent applicable to software... then any patent licenses granted by that Contributor to such Recipient under this Agreement shall terminate as of the date such litigation is filed”)
Nothing New Under the Sun

Defensive termination provisions of the scope in the Facebook grant are very common in patent licensing, outside of the open source landscape. Most patent licenses terminate if the licensee bring patent claims against the licensor. The reason is that a licensor does not want to be unilaterally “disarmed” in a patent battle. Most patents are only used defensively — asserted when a competitor sues the patent owner. A sues B and then B sues A, resulting in mutually assured destruction. If B has released its software under an open source license without a broad defensive termination provision, B is potentially without recourse, and has paid a high price for its open source code release. A gets to simultaneously free ride on B’s software development and sue B for patent infringement.

Finally, the Facebook grant itself is not new. The grant was released in 2013, and ReactJS’ popularity has been growing since then. As with many open source licenses, the industry’s willingness to absorb a new license depends on the tastiness of the code released under it. In the case of ReactJS, the code was great, and the patent license terms were new, but reasonable.

Is it Open Source?

Some have suggested that the BSD+Patents Clause violates the Open Source Definition. The OSD does not allow licenses that discriminate against persons or groups, or fields of endeavor. But the patent grant does not have license scope limitations; it terminates if the licensee misbehaves — that misbehavior having a lower threshold for actions against the code author than for others. So it seems likely that BSD+Patents does not violate the OSD, and moreover, CPL is already approved by the Open Source Initiative as compliant. CPL, like BSD+Patents, sets a lower threshold for termination based on patent suits against the code author.

What is the Upshot?

The practical result of the Apache Foundation’s decision is unclear. Category X licensed code cannot be included in an Apache Foundation repository. (That category also includes licenses like GPL.) Apache’s re-classification doesn’t mean anyone is restricted from using ReactJS —
it just can’t be committed in an Apache project. It’s not even clear that an Apache project cannot contain a dependency on BSD+Patents licensed code.

Meanwhile, in private business, there is little controversy about using code under the BSD+patent terms. Most companies have examined the marginal legal risk of this license compared to others (like Apache 2.0) and considered it underwhelming. Unless a company decides to sue Facebook (or accuse its products), the termination trigger has no actual effect. If you want to fling patent claims at a company that developed and released a great piece of code, removing the code from your business seems like a reasonable price to pay.

Some of the controversy seems to arise from concern that Facebook is advantaged over others in the license terms. But that is not the same as harming the open source community. The BSD+patents grant establishes the same “patent commons” as Apache 2.0, as a baseline, but provides more protection for the contributor (Facebook) against software patent claims of licensees. It’s odd that a community so opposed to software patents would find this objectionable, particularly in light of the array of defensive termination provisions that have been used in the past.

PLEASE NOTE: This blog entry is about the BSD+patent license, not about Facebook. This post represents my personal views only, and not the views of Facebook. I do represent Facebook on open source matters, but I did not draft the BSD+patents license grant.

SHARE THIS:

Like
Be the first to like this.

RELATED:

Google, WebM and BSD
Test your knowledge of open source licensing
Android Patent Peace (PAX)

AUGUST 19, 2017

Patrick McHardy and copyright profiteering

Many developers in the Linux community have concerns about the activities of Patrick McHardy. Here are answers to common questions.

24 Aug 2017 | Heather Meeker (/users/hmeeker) | 17 | 5 comments
Many in the open source community have expressed concern about the activities of Patrick McHardy in enforcing the GNU General Public License (GPL) (https://www.gnu.org/licenses/old-licenses/gpl-2.0.en.html) against Linux distributors. Below are answers to common questions, based on public information related to his activities, and some of the legal principles that underlie open source compliance enforcement.

Who is Patrick McHardy? McHardy is the former chair of the Netfilter core development team. Netfilter is a utility in the Linux kernel that performs various network functions, such as facilitating Network Address Translation (NAT)—the process of converting an Internet protocol address into another IP address. Controlling network traffic is important to maintain the security of a Linux system.

How much has McHardy contributed to Linux? This is not an easy question to answer. First, it’s not easy to assess the importance of contributions; all we can do is look at number and size of commits. And second, even if one tracks commits, the tracking mechanisms are not perfect. Git has a blame feature that tracks who nominally commits certain
lines of code to the git repository. Tools like cregit (https://cregit.linuxsources.org/) can be used with git blame to report commits at a more granular level of a code token, producing a more accurate picture of contributions at a file level. Git blame and cregit are useful because they both use publicly available information—the information just needs to be interpreted properly.

An analysis of blame with cregit can help assess McHardy’s potential contributions. For example:

- The bulk of his contributions appear to be concentrated during the period 2006-08 and 2012.
- Of approximately 135 files in which McHardy included his copyright notice, only 1/3 are files to which McHardy contributed 50% or more of the file’s code.
- His contributions appear to constitute well under .25% of the code in the kernel.

Most of McHardy’s contributions appear to be to Netfilter; however, blame might not always tell the whole story. For example, a committer can check in many lines of code having made only minor changes, or can check in code written or owned by others. For these reasons, the authorship of a committer can be under- or over-reported.

Records of contributions to the kernel prior to 2002 are not useful to identify contributors, because at that time, Linus Torvalds checked in all code. Patrick McHardy’s contributions did not begin until 2004.

The difficulty of establishing copyright ownership using development repository metadata arose in the Hellwig v. VMware case (https://www.theregister.co.uk/2016/08/15/vmware_survives/gpl_breach_case/). Courts may be reluctant to accept such information as evidence of authorship.
What copyright rights does McHardy have in the Linux kernel? Copyright ownership in large projects such as the Linux kernel is complicated. It's like a patchwork quilt. When developers contribute to the kernel, they don't sign any contribution agreement or assignment of copyright. The GPL covers their contributions, and the recipient of a copy of the software gets a license, under GPL, directly from all the authors. (The kernel project uses a document called a Developer Certificate of Origin, which does not grant any copyright license.) The contributors' individual rights exist side-by-side with rights in the project as a whole. So, an author like McHardy would generally own the copyright in the contributions he created, but not in the whole kernel.

What is "community enforcement"? Because the ownership of large projects like the Linux kernel is often spread out among many authors, individual owners can take enforcement actions that are inconsistent with the objectives of the community. While the community may have a range of views on how best to encourage adherence to the GPL's terms, most agree that enforcement should be informal (not via lawsuits) and that the primary goal should be compliance (rather than penalties). Software Freedom Conservancy (https://sfconservancy.org/), for example, has issued certain principles of community enforcement (https://sfconservancy.org/copyleft-compliance/principles.html), which prioritize compliance over the pursuit of lawsuits or money damages. There is no bright-line rule for when informal actions should become lawsuits, or how much money an enforcer should request. Most developers in the Linux community, however, consider lawsuits only the last resort, and are willing to refrain from legal action and work with users who sincerely wish to comply.

Why have so many open source lawsuits been filed in Germany? Some plaintiffs seeking to enforce open source licenses have filed their claims in Germany's court system. There are a few instruments for pursuing legal action in Germany that don't have exact analogs in the U.S. or other common law countries.
**Abmahnung** ([https://en.wikipedia.org/wiki/Abmahnung](https://en.wikipedia.org/wiki/Abmahnung)) ("warning"): The "warning" is a request from the claimant to the defendant to stop doing something. In the copyright context, it is a letter from the copyright owner requesting that an alleged infringer stop infringing. These letters are issued by lawyers, not courts, and are often the first step in a copyright enforcement action in Germany. In the U.S., the closest analog would be a cease and desist letter.

**Unterlassungserklärung** ([https://de.wikipedia.org/wiki/Unterlassungserklärung](https://de.wikipedia.org/wiki/Unterlassungserklärung)) ("cease and desist declaration" or "declaration of injunction"): The "warnings" will often have a "cease and desist declaration" attached to them. This "declaration" is like a contract—signing it will subject the defendant to legal obligations that might not otherwise exist. In particular, the declaration may contain obligations that are not required by the GPL itself. In Germany, it is common for such a document to contain penalties for noncompliance. In the U.S., the analog would be a settlement agreement, but settlement agreements rarely specify the penalties for breach—and in fact, in the U.S., "penalties" may not be enforceable in contracts. The "declaration" is not a court order, but if the defendant signs it, it may gain the legal force of a court order. **So, signing them before seeking legal advice is often not a good idea.** There are other approaches to consider in dealing with a complainant who sends a cease and desist declaration, including proposing a revised declaration with lesser penalties or obligations. Further, because a cease and desist declaration may also contain a non-disclosure requirement, signing one of these documents may also create additional difficulties, such as restricting the ability to seek support from other defendants or to alert the community about the claimant’s assertions.

For details, see [abmahnung.org/unterlassungserklaerung/](http://www.abmahnung.org/unterlassungserklaerung/).
Einstweilige Verfügung

(“interim injunction” or “preliminary injunction”): The “interim injunction” is a court order that is like a temporary restraining order in the U.S. A defendant’s non-response to a “warning” or “declaration” can encourage a plaintiff to seek an “interim injunction,” although there is no requirement that a claimant send a “warning” before requesting an “interim injunction” from a court. Interim injunctions for copyright infringement can prescribe penalties of 250,000 Euro or 6 months’ imprisonment. In the U.S., in contrast, criminal penalties for copyright infringement are extremely rare, and must be pursued by the government, not private parties. Also, in the U.S., courts do not prescribe remedies for future possible infringement—they only order defendants to stop current infringement or pay damages. In Germany, interim injunctions are also available ex parte, meaning that a plaintiff can apply to the court without the defendant being heard, and they can issue without the defendant’s participation. If you receive a “warning,” and suspect that a request for an “interim injunction” might follow, there is a possibility to file a preemptive “opposition” with the court.

For details, see Abmahnung.org.

Widerspruch ("opposition" or "contradiction"): The “opposition” is an opportunity for a defendant to file an opinion with the court that an "interim injunction" is not justified.

For an example of a case in which this process took place, see this English translation of a German court order.

How many claims has McHardy brought? Due to the lack of publicly accessible records for many German court cases, it is difficult to...
determine the precise number of actions brought by McHardy. It has been stated that McHardy has approached over 50 enforcement targets. For details, see Source Code Control (https://sourcecodecontrol.co/gpl/) and 7 Notable Legal Developments in Open Source in 2016 (https://opensource.com/article/17/1/yearbook-7-notable-legal-developments-2016). That doesn’t necessarily mean 50 lawsuits—it probably means 50 demands threatening a lawsuit. But it is difficult to verify this claim with public sources. For details, see Litigation and Compliance in the Open Source Ecosystem (https://www.slideshare.net/blackducksoftware/litigation-and-compliance-in-the-open-source-ecosystem).

Why hasn’t the community stopped McHardy? Various members of the community, including Software Freedom Conservancy, have reached out to try to persuade McHardy to change his strategy, but thus far they have not been successful. The Netfilter project recently published a licensing FAQ (http://www.netfilter.org/licensing.html) addressing concerns about McHardy’s actions.

What can we do to stop McHardy and other copyright profiteers? There is no one answer to this question, and there may be no way to completely stop them. But here are some suggestions for what might reduce the number of copyright profiteers.

Strive to comply with open source licenses. There are plenty of resources to learn how to comply with licenses, and how to set up an open source compliance program at your company. For example:


• The OpenChain project (https://www.openchainproject.org/) publishes a specification for recommended internal processes for open source management.

Don’t sign an Unterlassungserklärung (https://de.wikipedia.org/wiki/Unterlassungserklärung) before seeking legal advice. As explained above, an Unterlassungserklärung can subject you to obligations and penalties that are not found in the GPL itself. Don’t cooperate with the profiteers. You can make yourself a harder target, and enlist the help of other targets in the community.

Support open source development. Authors should not have to resort to profiteering to make a living. Companies that use open source software should not expect open source developers to develop software for free; they should chip in to support important projects.

Learn to recognize a copyright profiteer. Be aware of the general differences between community-oriented GPL enforcement and copyright profiteering. Community-oriented enforcement generally aims to achieve GPL compliance through education and assistance, while respecting users’ freedoms. Profiteering, by contrast, may focus on poorly researched scattershot claims and the threat of legal action for purposes of financial gain. Be on the lookout for assertions that prioritize financial gain and set the stage for unreasonable damages penalties.

Make claims public. If you are the target of a profiteer, and have a choice to make the claims public, doing so might help both you and others by discouraging their actions. As members of the open source community, we all share a duty to speak out against profiteers who seek to burden the
community with allegations that can be resolved in more appropriate and less contentious ways.

Topics:  Law (/tags/law)

---

About the author

**Heather Meeker** - Heather Meeker is a partner in O'Melveny & Myers' Silicon Valley office. She advises clients on technology transactions and intellectual property matters. She is an internationally-known specialist in open source software licensing. She received the prestigious IP Vanguard Award for private practice from the Intellectual Property Section of the California state bar for 2016. Best Lawyers named her the IT lawyer of the year for 2018.

* More about me (/users/hmeeker)
* Learn how you can contribute (/participate)

---

Recommended reading

- Don't over-React to the Facebook patents license (/article/17/9/facebook-patents-license?utm_campaign=intrel)
- An economically efficient model for open source software license compliance (/article/17/9/economically-freedom?utm_campaign=intrel)
- We don't make software for free, we make it for freedom (/article/17/8/software-freedom?utm_campaign=intrel)

Court Upholds Enforceability of Open Source Licenses

May 3, 2017

The District Court for the Northern District of California recently issued an opinion that is being hailed as a victory for open source software. In this case, the court denied a motion to dismiss a lawsuit alleging violation of an open source software license, paving the way for further action enforcing the conditions of the GNU General Public License ("GPL").

Artifex and the Dual Licensing Model
Plaintiff Artifex Software, Inc. provides Ghostscript—an interpreter for the PostScript language and the Adobe Portable Document Format (PDF). This popular software is available under a choice of licensing options, under a model often referred to as dual licensing. This business model was pioneered by MySQL AB in the 1990s and later adopted by many businesses seeking to maintain a balance between developing free software and funding development with license fees.

Under its dual licensing model, Artifex offers Ghostscript under two sets of license terms. The GPL—most famous for applying to the Linux kernel—is an open source software license that allows free modification and redistribution of software, subject to making the source code available under GPL license terms, among other requirements. This "copyleft" rubric ensures that all recipients of the software have free access to the source code. Artifex also grants alternative licenses on proprietary licensing terms, for companies that prefer not to comply with the requirements of GPL.

Hancom, Inc. integrated Ghostscript into its software product. It did not seek a proprietary license, but it also did not comply with the conditions of GPL. Artifex then tried to convince Hancom to comply with the GPL conditions, and when Hancom refused, Artifex filed suit to enforce its rights and vindicate the GPL regime.

**The District Court Affirms That Artifex’s Case Can Proceed**

Artifex’s complaint asserts that Hancom’s violations of the conditions of the GPL constitute both copyright infringement and breach of contract. Artifex has requested remedies including compensatory, consequential, and statutory damages, as well as attorneys’ fees and costs. Artifex also sought injunctive relief barring further infringement by Hancom and requiring Hancom to comply with obligations under the GPL.
Hancom moved to dismiss Artifex’s complaint on several grounds. The District Court denied Hancom’s motion to dismiss on each ground. A few aspects of the decision are of particular interest to the open source community. For example, Hancom argued that Artifex could not plead breach of contract for violation of GPL and could not request specific performance of the terms of GPL. Hancom also argued that copyright damages were not available because the GPL grants royalty-free rights.

Open Source Licenses as Contracts and Damages for GPL Violations

As part of its motion to dismiss, Hancom argued that using open source code offered under open source licensing terms does not form a contract. Whether open source licenses can be contracts in addition to conditional licenses has been an unsettled area of law. In the seminal case on enforcement of open source licenses in the United States, *Jacobsen v. Katzer*, the Federal Circuit Court of Appeals held that open source violations could be brought as copyright claims, but did not foreclose the possibility of bringing contract claims as well. In *Artifex*, the District Court ruled that Artifex’s breach of contract claim could proceed, finding that the GPL, by its express terms, requires that third parties agree to the GPL’s obligations if they distribute the open-source-licensed software.

Hancom also argued that because the grant of license under the GPL is royalty free, Artifex could not plead damages for a violation of the GPL. Relying on *Jacobsen v. Katzer*, the District Court concluded that royalty-free licensing under open source conditions does not preclude a claim for damages.

Implications for Open Source Software Licensing

Here, in denying a motion to dismiss, the District Court only holds that the claims may proceed on the theories enunciated by Artifex, not necessarily that they will ultimately succeed. Still, the case represents a significant step forward for open source plaintiffs. Many open source compliance claims have
been brought as copyright infringement claims, and *Jacobsen* affirmed this approach. Generally, copyright claims may afford plaintiffs more damages and stronger remedies than contract claims. However, contract claims may help a plaintiff pursue a violator’s worldwide conduct in a way that jurisdictional limits on copyright claims might not allow. Breach of contract claims may also be able to address reputational harm and other indirect non-economic benefits that a plaintiff might derive from enforcing open source license conditions. A breach of contract claim might also, in certain instances, allow for specific performance of open source obligations.

In the past decade, while enforcement of open source licensing violations has become more common, few enforcement cases result in published law. The open source community will be watching this case carefully, and this initial decision vindicates the rights of the open source authors to enforce GPL terms on both breach of contract and copyright theories.


2 535 F.3d 1373 (Fed. Cir. 2008).

This memorandum is a summary for general information and discussion only and may be considered an advertisement for certain purposes. It is not a full analysis of the matters presented, may not be relied upon as legal advice, and does not purport to represent the views of our clients or the Firm. Darin Snyder, an O’Melveny partner licensed to practice law in California, Melody Drummond Hansen, an O’Melveny partner licensed to practice law in California, Washington, D.C., and Illinois, Heather Meeker, an O’Melveny partner licensed to practice law in California, and Jesse Koehler, an O’Melveny associate licensed to practice law in California, contributed to the
content of this newsletter. The views expressed in this newsletter are the views of the authors except as otherwise noted.

Portions of this communication may contain attorney advertising. Prior results do not guarantee a similar outcome. Please direct all inquiries regarding New York's Rules of Professional Conduct to O'Melveny & Myers LLP, Times Square Tower, 7 Times Square, New York, NY, 10036, Phone:+1-212-326-2000. © 2016 O'Melveny & Myers LLP. All Rights Reserved.

Key Contacts

Darin Snyder
San Francisco
D: +1-415-984-8846

Melody Drummond Hansen
Related Practices

Intellectual Property & Technology