BRINGING FOSS INTO THE ENTERPRISE: THOUGHTS ON "BEST PRACTICES"

By Gwyn Firth Murray±
Matau Legal Group
www.mataulegal.com

Only a few years ago there was limited information available about how to manage Free and Open Source Software, or "FOSS", in the enterprise – either from a business or a legal perspective. Lots of "Fear, Uncertainty and Doubt", or "FUD", was gravitating around about how to manage the inflow of FOSS into proprietary technology companies. Initially, my response to questions about "best practices" for managing the inflow of FOSS into one's company – and the response of many of my colleagues also familiar with FOSS issues – was "just do what you already are doing; that is, leverage your existing intellectual property policies." The problem with that response was that it quickly became clear that many companies had no policies or procedures in place for managing and/or monitoring the inflow, and corresponding outflow, of software IP rights. Or, if they did, they weren't following them, regardless of whether those policies and/or procedures applied to "FOSS" or "closed source" software.[1]

Software is Software

So my first point here is that "Software is Software". In other words, companies should have policies and procedures in place for managing the intake of software and intellectual property rights. And then they should follow up and maintain them. In terms of such policies and procedures, there is a lot that is <u>not</u> new when it comes to FOSS. When considering what third party code they will bring in, commercial entities should be thinking about how that code fits within their overall business plan and strategy, future product line, distribution and revenue model. With those considerations in mind, companies should then carefully select those vendor products required or useful to implement that particular product development and marketing strategy. Companies should be doing a technical review and evaluation of potential incoming third party code for suitability within their own product; and reviewing (and/or negotiating) the applicable license(s) to be sure that they have the rights they need throughout the evaluation, development, production and distribution processes. And, of course, every company should have systems in place to monitor and enforce compliance with those license terms, and to track software acquisition and development, throughout all phases of technology development, production and release.

What might an effective intellectual property management program for managing the influx, use and outflow of software look like? Elements of such an ideal program would include:

- The company's executive team and all relevant company functions (i.e., legal, engineering, human resources, product management) not only understand the implications of software functionality and associated license rights in the context of the company's business, but also are clear about the company's approach to managing intellectual property rights.

- Communication between the legal and engineering teams relating to the use of incoming software is open and frequent.
- The company has a defined policy on incoming software that covers both proprietary and open source code in light of the company's culture and business. This policy provides clear guidelines to employees (and contractors) about when and for what purposes different vendor products, including competing technologies and FOSS, can and cannot be used and when legal review is required or not.
- When in-licensing software and throughout the development and production process, the company consistently asks and considers questions such as: are we bringing this software in for evaluation purposes only? For internal development and testing? Do we intend to modify this particular software? How do we expect to link it (i.e., statically or dynamically) or otherwise combine it with other company and third party software? Are we acquiring it for internal operations purposes only? For inclusion in a product that will be licensed to end users (or via resellers or Original Equipment Manufacturers (OEM's)) on a royalty basis? Will it be used to run a back-end server supporting an ASP (Active Server Page) or other kind of services business? Do we need "downstream" rights to sublicense or further distribute? The answers to these questions help determine what inbound license terms are and are not accepted by the company.
- The company has in place a process whereby all vendor/third party code receives an appropriate level of technical and legal review for compliance with license terms and license compatibility throughout the software development and production process, including evaluation/testing, development, production and distribution (including "downstream" licensing or redistribution).
- The company has clear policies covering the use of third party code by outsourced developers and other vendors, including representations and warranties required from such vendors. These have been communicated to all company vendors as well as to the company's purchasing and contract management organizations in advance of contract negotiation.
- The company's IT resources are engaged so as to create a central repository for third party and "home-grown" code, and the company has purchased, or designed, a software tool to conduct and maintain a dynamic inventory of all software used by the company. This software tool tracks where and how the software is used, when and how it is modified, applicable in-bound license terms, and assists the legal team in assessing license compliance and compatibility.
- The company has designated and publicized internally those individuals responsible for handling questions and approvals regarding the inflow, use and outflow of software code.
- The company has an intellectual property/software review board that consists of legal, engineering, human resources and product management staff that meets regularly and that is responsible for implementation, periodic review and enforcement of the company's IP policy.
- Company policy and accompanying practices are periodically reviewed and consistently enforced over time.

Working Well With FOSS Requires Cultural Change

All of the above recommendations apply equally whether one is operating within the world of "closed source" software or in the world of FOSS. However, the second main point I have to make here is that working effectively with FOSS requires cultural change – among companies, their employees and their lawyers. There are vast cultural and practical differences between working with and within the FOSS "Community" and that of "closed source" software.[2] The chances of implementing an effective policy are increased when we as lawyers are willing to learn about FOSS, get "down and dirty" by understanding technical specifications and software functionality, and adapt to the particular cultures of our client companies and the FOSS Community.

Licensing Isn't Just for Lawyers Anymore

Whereas lawyers used to be company "gatekeepers" without whose approval no software could come in the door, now engineers can – and do – surf the web and freely download to their computers at work and at home a wide variety of software, which comes in as both source and object code, and which is governed by a huge potential variety of license terms. When software comes in as source code, developers' ability to modify and create derivative works from it increases, which means it is easier to commingle with proprietary code – including a company's "home-grown" code and third party code. The consequences of such commingling – particularly with respect to FOSS code governed by "copyleft" license provisions such as those included in the GNU General Public License (the "GPL"[3]) – can be severe, such as triggering requirements to make company and other proprietary source code publicly available upon distribution of commingled code.

To be successful as a lawyer in this new world of FOSS, it is essential that lawyers for commercial entities recognize that there is no more ivory tower. Many software developers know – and even more think they know – much more about FOSS software and licensing than the average high tech company lawyer. Until recently, such software engineers have enjoyed a lot of freedom to download, use and play with FOSS without questions or interference from lawyers. They also have been writing licenses, sometimes without any input from the legal profession, and sometimes those licenses include technical restrictions that the average lawyer won't understand at first glance. Many developers have been working with FOSS a lot longer than we lawyers have.

Understandably, therefore, developers may resist lawyers' efforts to intervene and create policies that restrict their freedom to play with the exciting variety of software out there, or to release software they develop on the terms they choose. This places a burden on lawyers to build relationships with the engineering community (and particularly the FOSS Community) as never before. Given the vast array of software available and the sometimes bewildering license terms under which it is made available, lawyers wanting to learn about FOSS will need to ask for help from developers if we are to learn about available FOSS technology and interpret the associated variety of sometimes quite technical licensing terms. Being an effective lawyer in the world of FOSS requires a high degree of technical knowledge; start asking questions and asking for help, and you are likely to learn a lot. Again, if you demonstrate that you genuinely are interested in technology being brought in and developed, and that you are open to the company's use of FOSS and engagement of the FOSS Community, you likely will find that engineers are more receptive to your involvement.

In other words, we need to relinquish our role as "gatekeepers" that control the influx of intellectual property into our organizations, and recognize that FOSS requires that we engage in more open and collaborative lawyering. Rather than working alone in our offices or cubicles, we must walk down the hall (or fly around the globe) to work side-by-side with colleagues across functions and across borders. Face-to-face contact can be very important here. If you take time to walk around the engineering departments and/or set up brown-bag lunches or other meetings with engineers to discuss FOSS issues, you start dispelling the notion that lawyers are the distant enemy.

Another reason for making friends with software engineers is that the amount of software available, and corresponding number of licenses out there to consider, is truly daunting. When I checked recently, the number of FOSS projects hosted on sourceforge.net was more than 130,000, with the number of registered users listed as well over 1.4 million.[4] And Sourceforge is not the only place where FOSS is found. FOSS and other freely available software can be downloaded from other open source software repositories, commercial proprietary software companies (such as Microsoft, Sun and Oracle)[5], open source projects (such as JBoss[6] and PostgreSQL[7]), university and personal websites, and commercial open source companies (such as MySQL[8]), among other sources. While many of these software projects release their software under relatively well-known and well-understood licenses such as the GPL or Berkeley Software Distribution license (BSD license), many of them release software under different license terms that are referred to here as "FOSS" but may not meet accepted definitions of "open source".[9] Some software is downloadable without a clear path to the applicable license terms, or possibly with no license terms at all. The Open Source Initiative currently lists close to sixty "OSI certified" licenses,[10] but many software developers have chosen to issue their software under terms they have crafted themselves or created by modifying existing licenses – and that do not appear on the OSI list. And many of these developer-written licenses get very technical. For example, one must understand how the particular applications work together, how the programs "link" to libraries, how the software is written and compiled – in order to understand what you are permitted to do under many FOSS licenses.

But don't be scared. The same multitude of different licenses exists in "closed source" land as well – it's just that the terms aren't out there for everyone to see, and those terms generally have been written by other lawyers and sound more familiar to us as "legalese". Further, though "closed source" licenses have companies and their lawyers behind them, they generally don't have a passionate community around them to help back up enforcement efforts. In contrast, the FOSS Community sees as part of its mission the interpretation and enforcement of FOSS licenses, and this interpretation and enforcement takes place via often very blunt dialogue on public websites. So, here again, building and maintaining relationships with FOSS developers and the FOSS Community can be helpful. If you and your company are known to the FOSS Community as responsible licensees that endeavor to comply with FOSS license terms, the "flames" posted on FOSS-related websites may be kinder and gentler.

You Need to Read – and Re-Read – Each and Every License

So how do you deal with this daunting variety of highly-technical licenses? For a start, read the ones that apply to your software product, and read them carefully. If you don't understand what they mean, or what they permit, find your closest friendly software developer or IT professional and ask for help understanding how the software works, how it links, how it compiles, and how it is distributed from a technical perspective. Ask "stupid questions" and you, likely, will find out that your questions aren't so stupid. Ask other lawyers how they have handled the same kind of situation before and, likely, you will find that they are struggling with the same or similar issues. Post a question on the general discussion list at *www.open-bar.org* and see what comes back. If it's a "flame", push back on the sender and then contact the Open Bar site administrator or me. One of our key goals at Open Bar is to promote open dialogue among lawyers and legal professionals.

In my experience, most companies think the GPL is their worst nightmare because it is both widely-used and has strong "copyleft" attributes. And they are afraid of the GPL with some reason, particularly as the license is now under revision and new and different terms may apply to software licensed under that future version.[11] But, again in my experience, the GPL (at least in its current form) is not the FOSS lawyer's greatest problem. Most software developers by now understand the "copyleft" implications of the GPL and how to effectively navigate around those.

I find that more companies get into trouble with respect to other licenses, whether FOSS or "closed source." And the main reason they do is because they have not had a process in place for keeping proper records of which license applies (i.e., what license was in effect on the date the software was downloaded?), where that license is (is a copy still available on the web? Does the company have a hardcopy stored?), and making sure that the permissions under the license continue to match the company's needs through out the development, production and distribution process. More and more engineers understand the implications of downloading FOSS software and have read – and care enough to abide by – the terms of both their employment agreements and the applicable license before they do so. However, even the best-informed and best-intentioned developer may forget about license terms that permitted testing and evaluation but prohibit modification or distribution by the time they start incorporating the third party code and using it to develop product. Again, "Software is Software", and here is where a well-thought-out IP management program helps keep you out of trouble.

Other areas where I see companies get into trouble include forgetting about outsourcing and not considering what "distribution" means under the terms of a given license. Companies may be diligent about their own intake of third party software but then fail to monitor such intake and use by the outsourced consultants they have hired to develop their software. Companies should have in place - and purchasing and contract management staff should be familiar with, and equipped to enforce - policies and requirements around vendors' use (or non-use) of FOSS in any products or services coming into the company. Similarly, companies should be alert to the question of what might be considered "distribution" within their own entity or to agents acting on their behalf. For example, some licenses may explicitly provide that offering software on an ASP basis equals distribution, thus triggering the reciprocity requirements of that license. Even if the license is silent on whether running an ASP service equals a "distribution", if a multi-national corporation has affiliated companies and uses "back-end" software developed at one affiliate to provide consulting services or run an ASP offering at a sister company, is that a distribution? Does this analysis change if the affiliated company is a joint venture that is only partially owned by the parent? What about the company's use of outsourced consultants to develop its software? If software is sent back and forth between the outsourcer and the client company during the development process, is that distribution?

Asking these questions may not lead to clear answers in any particular case. However, having good relationships and open communication with FOSS developers and other lawyers will only help you when you need to "brainstorm" and come up with creative solutions.

Conclusion

There are valid reasons that "Fear, Uncertainty and Doubt" linger around the use of FOSS in the commercial enterprise. Working with FOSS requires that we acquire more technical knowledge and be more diligent about implementing and maintaining "best practices" programs than we may have in the past. And it also means that we must build relationships with our clients and other lawyers as never before. If we are to be effective, we need to be open to help from developers who may have more in-depth knowledge of FOSS technology and licensing terms than we do. To figure out answers to legal questions when almost no statutory or case law on FOSS exists, we need to be open to brainstorming with other lawyers – even if they are our competitors. But what goes around ultimately does come around, in my view. When we are open to asking for help and to learning from the FOSS Community and from each other, the FOSS Community – and our fellow lawyers -- likely will be more open to working with, and learning from, us. In sum, Open Source requires Open Lawyering.

Footnotes

± Gwyn Firth Murray was one of the "early adopters" in the legal community of a focus on open source software. She is founder and principal of the Matau Legal Group, which offers a broad range of commercial, licensing, and other legal services to both start-up and established companies in the high tech and biotech industries (see www.mataulegal.com). Ms. Murray is co-founder of Open Bar, Inc., a not-for-profit organization focused on legal rights and responsibilities in the world of open source software (see www.open-bar.org) and currently serves as Vice-Chair of the Open Source Committee of the American Bar Association Section of Science and Technology. Ms. Murray is a graduate of Stanford University Law School and also holds an M.A. in Latin American Studies from Stanford University. She obtained her B.A. magna cum laude and with distinction in economics from Yale College.

1 *See generally* Bradford L. Smith and Susan O. Mann, *Innovation and Intellectual Property Protection in the Software Industry: An Emerging Role for Patents?*, 71 U. Chi. L. Rev. 241 (Winter, 2004).

2 For more information on this topic, please see the following papers written by the author: "Free and Open Source Software: An Introduction" (January 2006) and "Getting with the Program: A Guide for Lawyers Working with Open Source Software in the Enterprise (October, 2005), both available at www.mataulegal.com and www.open-bar.org. This article is derived in large part from the latter paper.

3 See http://www.gnu.org (accessed on October 25, 2006).

4 See http://sourceforge.net (accessed on October 12, 2006, when it listed 131,830 Registered Projects and 1,412,096 registered users).

[5] See microsoft.com, sun.com and oracle.com generally.

[6] See *http://labs.jboss.com/portal* (accessed on October 25, 2006).

[7] See *http://www.postgresql.org* (accessed on October 25, 2006).

[8] See *http://www.mysql.com* (accessed on October 25, 2006).

[9] Various definitions of what "open source" means exist. One of the most commonly referenced is that of the Open Source Initiative, which can be found at *http://www.opensource.org/docs/definition.php*. Readers may also be interested in the discussion of the difference between "open source" and "free software" found at *http://www.gnu.org/philosophy/free-software-for-freedom.html*.

[10] *http://www.opensource.org/licenses* (accessed on October 12, 2006).

[11] I believe that this topic is covered in another article in this newsletter, so I will not go into detail about it here.