LINUXWORLD™ MAGAZINE
WWW.LINUX.SYS-CON.COM

Dual-Licensing Open Source Business Models

Mixing proprietary terms with the GPL

By: Heather Meeker

April 6, 2005

To Open Source or not to Open Source - that is the question. Or is it? Open Source has matured into a robust development model, and many businesses that shied away from it are reconsidering it. The good news these days is that Open Source is no longer an all-or-nothing choice.

What Is a Dual-Licensing Model?

A dual-licensing model is a business model in which a company that markets a commercial software product gives its licensees the choice of two licensing models: Open Source and closed source (or "proprietary"). In this business model, a customer can choose to license the software under the terms of an Open Source license such as the GNU General Public License (GPL). Open Source licenses let licensees sublicense the product's source code to multiple levels of sub-licensees, but require re-licensing in source code format, usually on identical terms. Alternatively, the customer can choose a closed source or proprietary license with more conventional licensing terms that limit his ability to re-license the product, or restrict him to object code sublicenses only.

The theory is that the dual-licensing model helps both the free software community and the commercial software licensee. The open availability of source code lets the software be improved by those who have the right to change it under an Open Source license. The proceeds from commercial licensing help fund additional development, and help establish the product as a commercial standard.

Who's Doing It?

Several companies have broken ground by launching dual-licensing models.

The best known is mySQL AB, which produces an Open Source database. mySQL has two license options: a commercial license and a modified GPL license, which allows the licensee to distribute mySQL code under GPL side-by-side with other FLOSS licenses. Commercial licensees get a commercially supported product with a level of assurance from mySQL that doesn't require that their mySQL-based software be Open Sourced. mySQL can offer a commercial license because it owns the mySQL code. The commercial license sells for several hundred dollars per server. The two licensing models cover identical products.

TrollTech AS uses a dual-licensing model with its QT product, a C++ application development framework. It charges a per-developer price for its commercial license. The two licensing models cover identical code bases. The free licensing model is available under the GPL (or the QPL, a free software license that's not that widely used).

Active Endpoints, Inc. uses a different strategy, having launched a separate Open Source product that complements its core product. Its ActiveBPEL engine is a runtime environment for executing processes based on the BPEL4WS 1.1 specification. It is the core of Active Endpoints' ActiveWebflow family. Active Endpoints licenses the Active BPEL Engine under the GPL, and its suite of products on commercial licensing terms.

Mandrakesoft distributes a popular Linux distribution. It, too, has multiple license options, but its commercial product is different from its free software. Mandrake's Web site says that "The difference is not the license of our products, but the licenses of third-party software contained in the commercial distributions. Download edition Mandrakesoft products only include components licensed under a Free Software/Open Source License. Commercial products may also include software add-ons that are covered by a proprietary license that often prevents free copying and redistribution of the product."

SleepyCat Software distributes Berkeley DB, a data storage and retrieval development package, under multiple licenses. Its Open Source option requires any redistribution to include source code. It also offers commercial licenses: "If you do not want to release the source code for your application, you may purchase a license from Sleepycat Software," it says.

Many Open Source projects offer licenses under multiple Open Source options - such as the Mozilla Foundation, which gives its licensees the choice of Mozilla, GPL, or LGPL.

IP and Licensing Concerns

Lurking in all of this are some profound legal issues:

- ***Ur-Licensing***: The decision "to Open Source or not to Open Source" is only possible for the original author of software code - the "Ur-licensor." This is because, once software is licensed under a "viral" or free software license, it must forever be licensed under those terms. No licensee can change this. The necessity of being the original author is borne out by the dual-licensing businesses described above. All that offer commercial licenses are the authors of their code - but for Mandrakesoft the exception proves the rule. Its commercial licensing option is dictated by third-party licensing terms.
- ***Contributions***: The premise of dual licensing is that the community can contribute to the product's code base to build better code. So, assuming a company has the right - by virtue of having written its code - to give its licensees a choice of Open Source or commercial terms, it must decide how to manage contributions from the Open Source community. There are two ways to go about it: by requiring contributors to assign all rights to their contributions to the company, or to require the contributors to grant a license to the company that's broad enough to allow redistribution on both commercial and non-commercial terms. Either approach can meet with significant resistance from the contributor community, so public relations - and a good licensing FAQ - are key.

- ***Trademarks***: Any company launching any business initiative needs to consider branding. Open Source businesses are no different, but the trademark policies covering Open Source products can be trickier than for proprietary software. The problem, in a nutshell, is that trademark law and Open Source are fundamental opposites. Trademarks are only as valuable as their owners make them by controlling the quality and source of the products that bear them. Losing control means losing rights through a process known as dilution or blurring. But Open Source is all about lack of control. A company can decide to grant the right to modify its software to all and sundry. But it must decide how far to let those modifications stray from the official release before the product's branding has to be changed. Different Open Source companies and projects approach this issue differently. Some allow limited use of the mark associated with the product; some allow none. A savvy dual-licensing model requires serious thought about trademark policy, as well as active policing of the policy to avoid compromising of a company's trademark rights.
- ***Patents***: Finally, any company that offers Open Source licensing terms for code it has written must consider the effect on its patent portfolio. Releasing code under Open Source licenses can give licensees an implied license to a patent that covers the code - though the scope of that license will be unclear. Some of the more "corporate" Open Source licenses include express patent grants, and this helps clarify the scope of the patent licenses that are included. But any commercial enterprise that releases Open Source code must make this decision with its eyes open after formulating a general patent strategy.

Corporate Concerns

Companies pursuing a dual-licensing strategy should consider the lesson - learned too dear and too late by many start-ups - that distinct businesses are better separated into different operating entities. While many in the business community are embracing Open Source as a development method or a business model, some still consider it a substantial negative. A start-up may find that a potential acquirer or investor dislikes the idea of buying or investing in Open Source assets, or, at least, doesn't value those assets seriously. If your acquirer tells you that it loves your product, but doesn't want to be involved in Open Source, you may need to segregate the Open Source business prior to sale. Doing this on the eve of an acquisition can be difficult, as any CEO who's gone through a spin-off can tell you. The longer a business operates, the harder it is to segregate its multiple business lines. Assets like copyrights, trade secrets, trademark rights, and goodwill can be difficult to un-mingle. The solution may be to create dual corporations to handle your dual-licensing model.

Creating dual corporations also helps compartmentalize liability, so your for-profit business won't be responsible for losses or liabilities that the Open Source project may incur. But to preserve this advantage, you'll need to follow corporate formalities like directors' meetings and corporate minutes, or the "corporate veil" that limits liability can fail.

If you decide to use a dual-corporate structure with your dual-licensing model, you might also consider designating one of them as a not-for-profit entity. Many Open Source projects operate through public benefit corporations (classified under 501(c)(3) of the Internal Revenue Code). However, before doing this, you should consult a tax attorney or an advisor who's experienced in this area to establish this status and help set your expectations. Keep in mind that not-for-profit entities are obliged to operate for the benefit of the public at large, not to remove liabilities and costs from your for-profit balance sheet. Transfer payments between for-profit and not-for-profit entities can land you in tax trouble, and compromise your not-for-profit status. This status has other more important uses - getting donations and grants, and garnering goodwill with the community. It also lets you invite participation by board members who prefer not to sit on boards of private for-profit corporations.

Does It Work?

In a sense, the business theory behind dual licensing is nothing new. Software companies have historically offered "lite" versions of their products as teasers, shareware, or free evaluation copies. Familiar examples are the video game Doom (which was licensed for free in a version that contained only a few levels, whereas the full game contained many others), and the Netscape browser (which was licensed for free to users of the client portion, whereas the server portion was licensed to businesses on commercial terms). Hardware companies have traditionally included software - even software source code - free with their products to leverage hardware sales. This is sometimes called "widget frosting." For instance, a company that sells computer hardware may Open Source its drivers or interface tools, which are not normally sold separately to the customer. And of course IBM is a major proponent of Linux, which drives sales of its hardware, software, and consulting services.

So, there's no reason for dual licensing not to work as a business model. It's too early to tell, but the dual-licensing model seems to be serving the companies doing it - not only for financial success but to engender community support as well. Dual-licensing models are accepted by most members of the Open Source community, and increasingly by investors concerned with the bottom line.

About Heather Meeker
Heather Meeker is a shareholder at the Silicon Valley office of Greenberg Traurig, LLP, an international law firm well known for its intellectual property practice. She specializes in drafting and negotiating intellectual property transactions for software and other technology clients, with an emphasis on open source software. She is co-chair of the ABA committee to open source. Heather was a programmer/analyst before becoming an attorney.