Considerations for Trademarks, Nomenclature and FLOSS Communities

Patrick Finch[1]
1 Mozilla Corporation, Eskilstuna, Sweden patrick@mozilla.com, WWW home page: http://www.patrickfinch.net

Abstract. The use of trademarks is a critical, and frequently overlooked, aspect of the software industry. For FLOSS projects, the choice of name to be trademarked and the scope of that trademark are important decisions that should be made early in the life of a project. When choosing a name, consideration should be given to the project's mission, and how a community is expected to form around the project. This paper does not offer legal advice, but instead observations on how the formal use of names (as trademarks[1]) have helped or hindered FLOSS communities.

1 Introduction

In February 2008, Roy Fielding, one of the founders of the Apache web server project and one of the founding members of the OpenSolaris Governance Board resigned from the OpenSolaris project after Sun Microsystems, effective sponsor of the project but with no official capacity within the project's governance, oversaw the building of a binary distribution of the OpenSolaris source code and called it "OpenSolaris".

Fielding noted:

The truly stupid aspect of this issue is that, [as far as I can tell]*, most of the people still hanging around here (including myself) think that it is a good idea for OpenSolaris to produce a reference distribution of some kind within an open development project of the OpenSolaris Community...This well is poisoned;* [Sun

[1] This paper is not legal advice, and does not address the issue of whether or not a project should register a trademark, but we should note for the purposes of the discussion that the consistent and persistent use of a term which is not already used in a field of endeavour can come to constitute a trademark, just as the failure to ask other parties from using that term to describe their output can lead to that mark being invalidated. Similarly, there is also a concept of "fair use" with respect to trademarks in some jurisdictions. That is also not the focus for this paper. "Trademark" here, then , will be used to apply both to registered trademarks and the consistent and persistent use of a term with respect to the output of a project.

Microsystems] *has consumed its own future and any pretense that the projects will ever govern themselves (as opposed to being governed by whatever pointy-haired boss is hiding behind the scenes) is now a joke*[2].

In other words, what made the decision controversial with Fielding was not the outcome itself, which had broad agreement, but that the name by which the community identified itself was not under the control of that community.

However, it is not surprising that Sun trademarked the term "OpenSolaris", given the importance of trademarks in commercial software: any software which will be commercially distributed (even if that software is delivered for free), requires the use of a trademark to certify that software to. The name needs a point of control (a trademark owner), who, at any given moment, can define what that trademark actually represents in terms of versioning, test conditions and so forth. Although Sun did not appear to have a specific strategy for using the name "OpenSolaris"[3], it would also have been highly unlikely not to assert rights over the use of the term.

At the start of 2008, there was a brief debate within the open source community whether or not, in the wake of more and more software being distributed as a web service, open source licenses were still relevant and whether, in fact, trademarks were a more important control point for open source communities. This paper certainly does not argue that, but it does argue that just as trademarks offer protection for businesses, so do they offer protection for both commercial and non-commercial open source projects, and are a useful tool for clarifying the relationships within a community.

2 Trademarks and FLOSS Culture

FLOSS communities are famous for unapproachable nomenclature (in-jokes within development teams and contrived acronyms are both popular choices[4]). And while the use of trademarks by FLOSS projects may not receive the level of opprobrium of the use of software patents, trademarks do not have the same level of acceptance and familiarity as copyrights. As the Software Freedom Law Center puts it,

Way down in the roots of Software Freedom are the same ideas from which the belief in free speech derives. FOSS projects are not often built by people who value censorship, and there is a strong belief within most FOSS communities that projects should thrive on their merits, and not use legal weapons to silence critics and

---

2 http://mail.opensolaris.org/pipermail/ogb-discuss/2008-February/004488.html
3 http://lwn.net/Articles/139739/
4 http://lists.xcf.berkeley.edu/lists/gimp-user/2006-September/008696.html

competitors. It is for this reason that in many cases, projects will encourage use of their name even when commercial, proprietary software-producers would forbid it[5].

2.2 Mission

What is considered normal within a FLOSS project is to have a mission: this may be in the form of a business objective for an entirely commercial open source project, or it may be targeted at a specific objective. Such an inclusive and shared vision is widely considered a best practise in building a community. It is also revealing about what the project aims to achieve, and can indicate what output a project might seek to protect by trademark. Does a project intend to have derivative distributions, or a single distribution? Does it intend always to be conveyed under a copyleft license? Does it intend to support the business model of one company, or several? How should those business models be supported? These are all questions that ought to be understood before a decision on nomenclature is made, in the formative stages of the community.

3 Community Models

The ability to set expectations accurately, amongst those joining a community will be a contributing factor to the long-term sustainability of that community. Specifically, it is important to be able to respect and to accommodate the motivation of individuals who join a community.

Principally, there are three entities a name can be applied to: a code base, a distribution of that code base and a community that works on that code base. And while there is a tendency casually to describe a community by the software it is associated with, “the Linux community”, “the MySQL community” etc., the rights of community members in relation to that software will differ depending upon the ownership and control of the trademark.

3.1 Community Roles

For the purposes of analysis, we can employ a simplified taxonomy three kinds of FLOSS community member[6]. All add value to a project, all will have different motivations for involvement:

- *Developers* are what most commonly spring to mind when discussing the open source community – specifically, people who contribute code, either original code or patches or extensions. In many cases, developers will be adding to a code base to meet their own objectives (in the way that the Linux code base is contributed to by a number of competing companies).

[5] http://www.softwarefreedom.org/resources/2008/foss-primer.html#x1-600005

[6] With thanks to Simon Phipps of Sun Microsystems

- *Deployers* will, like developers, be necessarily technical, but unlike developers, deployers are not contributing code: they are attracted to a community by the ease of access (to software, to documentation, to original developers) that it offers (such as members of the MySQL community).
- *Users* of a project, who contribute by their very existence bug data etc., and in many cases, will add to a knowledge base for other users (the Ubuntu community is an example of a vibrant and contributing user community, amongst other things)

This taxonomy is a simplification, and contributions to open source projects also take the form of translation (a form of extension) and testing (a form of deployment) and so forth.

3.2 Business Models

The topic of "open source business models" can dissolve into an exercise in rectifying a category error -the possibility of a collaborative development methodology (open source) can relate to a business model in a number of ways, and some businesses built entirely on FLOSS monetize software-as-a-service (SaaS) models. What's more, many open source projects with viable communities do not have a business model associated at all, and may adopt a *laissez-faire* attitude to their trademark. However, for the purposes of analysis, we can divine three principle models:

- *Dual licensing*, (e.g. Sun's Java SE) where a code base is available under an open source license or, at a cost, a proprietary license (NB. many projects offer binaries under a different license to the code base, that does not mean that they operate this as a business model)
- *Services*, (e.g. Red Hat Enterprise Linux) where the business model is to add value to the user by maintaining the software and offering warranties etc. (often known as "subscription" software)
- *Indirect*, (e.g. Mozilla Firefox) where the software has a large user base and is a vector for services for other companies (in the case of Firefox, for companies offering search services).

4 Case Studies

4.1 The Linux Market

One key factor in the continued slow adoption of Linux on the domestic and institutional desktop is the difficulty in certifying hardware and software to Linux,

which is a code base and not a binary product. When one buys peripherals with the Linux mascot, Tux, appearing on the packaging, it is worth noting that his presence almost certainly signifies little other than that the device will "probably work". The emergence of well recognised branded Linux distributions – Red Hat Enterprise Linux, Ubuntu – represent the solution to this problem.

The provision of trademarks and commercial entities controlling those trademarks has enabled the growth of an ecosystem of software and hardware on Linux, which can be sold and supported to run in a known configuration, rather than with a disparate set of components.

Curiously, this effect has also protected the enterprise Linux market from existing enterprise software incumbents. In 2006, Oracle entered the enterprise Linux market with Oracle Enterprise Linux. While the product was comparable to Red Hat Enterprise Linux (it is based upon CentOS, which is in turn based upon Red Hat Enterprise Linux), the impact upon Red Hat's business has been minimal as little software was certified against the new entrant. The quality of Oracle's product was not the issue: the ability to run the product in a configuration that multiple parties recognised and supported, was.

Similarly, Ubuntu is increasingly visible as a consumer brand, and, as a trademark controlled by Canonical, Dell (and others) have been able to certify and ship hardware with Ubuntu pre-installed.

4.2 OpenSolaris

In 2005, Sun Microsystems launched the OpenSolaris community. "OpenSolaris" represented an embedded trademark (Solaris), although there was no intention at the time to create an OpenSolaris product. The hope was to attract other developers to extend the platform, while the Solaris Operating System would remain the platform Sun Microsystems would certify to. However, the absence of an "OpenSolaris" that new users and deployers could install and boot slowed adoption of OpenSolaris, leading Sun to create an OpenSolaris distribution in 2007. This appeared to represent a recasting of the OpenSolaris community from outside the governance of the project, resulting in considerable friction. OpenSolaris is now cast as a free desktop operating system similar to Ubuntu for deployers and users, as well as a code base for developers.

4.3 Mozilla Firefox

The Mozilla project had to rename its browser twice, after both Phoenix and Firebird were deemed trademark infringements – Firefox being the final choice. The Mozilla code base itself is widely licensed under the GPL, LGPL and MPL, but the name "Firefox" and its accompanying artwork are restricted to official Mozilla

Firefox builds. And while the Mozilla community has used the Firefox artwork in a wide variety of contexts (manga cartoons, crop circles), Mozilla remains quite rigorous in what binaries may use the trademarks. This policy, however, was found to be in conflict with the Debian Free Software Guidelines (DFSG), which lead to the IceWeasel fork of Firefox on the Debian platform (it is worth noting that the Debian project applies this rigour to its own marks, ensuring that there is a version of Debian artwork which is compliant with the DFSG). The Mozilla code base, meanwhile, retains a strong developer community, both contributing to Mozilla product releases and to other projects, outside of Mozilla's control.

4.4 Java

Another Sun open source community was launched in late 2006 around the Java SE platform, OpenJDK. The Java trademark was of critical importance to Sun Microsystems: the success of the platform rested upon "Java" having a consistent meaning across platforms, and securing licensee revenue: Microsoft had been forced into settlement a few years earlier for distributing a non-conformant version of Java, and open source implementations of the technology have carefully avoided using the name.

Extending the use of the Java name, then, appeared problematic. The approach taken by Sun to this problem was to "open source" the Java community mascot, Duke, under a permissive license[7]. This helped to give the OpenJDK community a strong "official" identity while preserving Sun's ability to pursue a dual licensing business model.

4.5 JBoss

JBoss (now owned by Red Hat) has faced criticism from people who consider themselves part of the JBoss community[8], because JBoss enforces trademarks both on code, but also on services offered against that software. Where an expert in the field of Linux (and therefore, by a loose definition, a "member of the Linux community"), might expect to offer professional services as a "Linux consultant", the same might not be true for an expert on a product that JBoss holds a trademark to.

---

[7] http://duke.dev.java.net/

[8] http://thejbossissue.blogspot.com/ http://bill.dudney.net/roller/bill/entry/1

5 Conclusions

For a Dual-licensed model, to attract a healthy community, other entities will want to be able to create their own distributions of the code base. It is probably the case that a project wants to attract other developers, and therefore the binary product should have a distinct identity to the community, or at least, to be able to differentiate the community from the binary product.

Communities that are built on service models are very interesting cases, and remain a grey area. Where one contributing company intends to to be the exclusive provider of services against a binary product, it probably should seek to extend trademark rights over both services and binary products. If it is then going to extend that trademark over the community too, the purpose of it being an open source community at all is called into question.

Projects which intend to have an indirect model will tend to be focused around a single binary. If a sustainable community is to exist with other developers in it, then it probably will look to create other branded binaries from the same code base, but with very distinct differences.

Two best practises do tend to emerge. Firstly, it is in almost all cases useful to distinguish between the software product and the developer community (both in cases where the community is producing multiple products, and when a business seeks to control the binary). Secondly, decisions about naming a project, its code base and resulting products should be made early in the life of a project. They should be made in line with what community model, and any attendant business model, is desired. This can serve to set correct expectations with potential community members, reducing the risk of disagreement later. Moreover, the kind of situations where a name change (or change in meaning) are being considered may be attended by high emotions and strong disagreements. It is better to set policy in advance of such circumstances.