EVALUATION OF OPEN SOURCE LICENSING MODELS FOR A COMPANY DEVELOPING MASS MARKET SOFTWARE

Mikko Välimäki, researcher
Helsinki Institute for Information Technology
P.O.Box 9800, 02015 HUT
Finland

Ville Oksanen, visiting scholar
School of Information Management and Systems
UC Berkeley
USA

ABSTRACT

Open source and free software products have challenged established licensing models for many types of mass market software products. Especially markets in operating systems, development tools, web servers and databases have significant open source and free software products. A software company that wishes to benefit from the non-technical innovations in open source and free software licenses has to carefully study the markets and characteristics of different open source licensing models. This article contains a preliminary evaluation of several frequently used open source licenses and licensing models from the perspective of a company developing mass market software products for competitive markets. Comparing them to commercial licensing models, the article aims to explain when and how open source licenses make economic sense.

KEYWORDS

Software licensing, open source, software distribution, software economics

1. INTRODUCTION

License choice is perhaps the most important strategic decision for mass market software producers. While custom software development may use case by case negotiated license and development agreements, mass market software licenses have for the most part standardized terms and conditions.

Lerner and Tirole [5] have listed relevant considerations in the choice of open source license. Their list includes fear of commercial hijacking, impact of software patents, incentives to produce complementary software and familiarity with the license. License choice generates externalities to end users, other developers and companies selling competing or complementary products.

The paper proceeds as follows. We start with necessary background discussion defining most popular licenses and relevant economic attributes describing mass market software. Then, we continue with license choice analysis under three separate topics: the implications of license choice to (1) copyright ownership, (2) development process and (3) product distribution in competitive markets. The viewpoint is of a software company that produces mass market software for competitive markets. Real life examples of licensing practices are used to illustrate more theoretical concepts discussed in the main text. The article ends with a general explanation of when and how open source licenses make economic sense

2. BACKGROUND

2.1 What are open source licenses?

Software products are licensed, not sold. Traditionally, software companies have developed software in-house and used various kinds of end user license agreements that give licensees limited rights to use the software for specific purposes. Usually, source code is not shared and distribution is restricted. Meanwhile, in academic circles software has been for a long time developed with the principles of open source code and free distribution.

Richard Stallman, a former staff member at MIT computer lab, published the first version of GNU General Public License (GNU GPL) in 1989 as part of the GNU project and it has since become the flagship of the so-called free software movement.[13] Stallman also introduced Library General Public License (LGPL), which was later renamed as Lesser General Public License. Another early open source license is called Berkeley Software Dis-

tribution (BSD), which was introduced with the BSD Unix software developed at the University of California at Berkeley.

Open Source Intiative (OSI) was started in 1998 to address increasing corporate interest in Linux and other software developed under open source principles. OSI has certified after its launch over 30 licenses, which comply with the general terms of the Open Source Definition.[10]

2.2 Most popular licenses

It is possible to make some approximations of the popularity of open source licenses by accessing data at development centers on the Internet. These percentages tell how popular particular licenses are in the number of products, not in the number of copies of the license in use.

At sourceforge.net, which as of July 2002 hosted close to 30 000 mainly open source software projects, the share of projects under GNU GPL is 67%, LGPL 10%, and category combining BSD, MIT, Apache and Public Domain 12%. The relative importances of Mozilla and Artistic licenses are also high since they cover essential products (Mozilla browser and Perl language) while their use in other projects is rare (less than 2% at sourceforge.net). Other licenses have minimal usage.

In this short article the focus is on the three most popular open source license categories. Also commercial licensing model is discussed for comparison reasons.

2.3 License categories

In terms of rights given to users, the most popular mass market software licenses can be categorized from most permissive to the most restrictive as follows:

- BSD, MIT, Apache – these licenses are *all permissive* allowing free distribution, modifying, and license change; from economic viewpoint, also public domain software (software with no copyright) falls into this category
- LGPL – allows free distribution, modifying and license change if bundled as a whole into new work; derivative works must be under LGPL or GPL (license is *persistent*)
- GNU GPL – allows free distribution and modifying but all bundled and derivative works must be under GNU GPL (license is *persistent and viral*)
- Commercial – allows the use of software only in specific circumstances and hence these may be called *all restrictive* licenses

These categories are presented with more detail in table 1 below:

License	GNU GPL	LGPL	BSD/MIT/PD	Commercial
Type	**Persistent and viral**	**Persistent**	**All permissive**	**All restrictive**
Popularity	**67% among open source**	**10% among open source**	**12% among open source**	**N/A**
Derivate works	**Only GPL**	**GPL or LGPL**	**No restrictions**	**Not allowed**
Bundling	**Only GPL**	**No restrictions**	**No restrictions**	**Restricted**
Patenting	**Free licensing required**	**Free licensing required**	**Not covered**	**Restricted**

Table 1. Most popular mass market licenses.

2.4 Mass market software

For the first, mass market software can be characterized by the intended users:

- *End-user* software with main differences in application and game software. While the value in games is based on one-time user experience, the value of application software is based on functionality.
- *Developer* software in which the value is in creating functionality to new works. Modifiability and adaptability increase the value of developer software.
- *Embedded* software is for third parties who wish to implement the software as a part of a larger product, either hardware or software. Compatibility is therefore necessary.

Second, main economic attributes that characterize software products can be classified as follows [6] [11]:

- *Marginal cost* of distributing an additional copy approach *zero*; costs are generated from development, marketing and support
- *Network effects* occur when the value to software users depend on the number of other users. Compatibility is therefore essential.
- *Lock-in* to particular software may arise because software requires compatibility from hardware and other software
- *Life-time* of software can be *indefinitive* in the case the software is incremental and new functionality can be added

3. OWNERSHIP OF RIGHTS

3.1. Why it matters

It is of essence for software companies to have undisputed rights to the software product it wishes to license. Ownership of rights is central because it allows company to

price its software, change its licensing policy and distribute software with different licenses.

A major legal risk in using open source licenses is that the license may "dilute" the ownership and even eliminate the possibility to relicense the software. Therefore, rights ownership must be managed carefully.

3.2. What are the rights?

In an efficient license contract, all known rights that cover the software product in question are addressed. Most important of the rights are copyright, trademark and patents. All creative software is covered by copyright and now also patents are granted to software products.[6]

However, there is a practical problem that proprietary rights in software are not clearly defined in copyright, patent and trade mark laws. Instead, there is ambiguous language in law, interpretation problems, clear gaps and overlaps. Therefore software licenses tend to be more or less incomplete.

This makes it more understandable why some licenses can be hesitant in addressing some specific rights. Persistent licenses GNU GPL and LGPL are in this sense significant because they require free licensing of any patent covered by the licensed product (copyrighted software). This is to say that *GNU GPL and LGPL licenses and software patents are incompatible*. A company cannot use at the same time a permissive license on a software product it has a patent upon and wishes to license the patent for a fee.

3.3 Who is the author?

The one who has written new or rewritten old software is granted exclusively copyright to the work. However, with multiple authors the copyright ownership may also become distributed, which poses challenges to licensing. We can think of three typical situations [1]:

- Distributed incremental development with no coordination. In this case every contributor has copyright to his contribution (*bundled work and authorship)*
- Focused and centrally controlled development. In this case, every contributor has copyright to the work as a whole (*joint authorship*)
- Complete rewriting of existing works. In this case the rewriter(s) have copyright to the new work overriding all previous copyrights *(new authorship*)

One difficult problem with distributed development is *employment relationship.* According to many national laws, the employer owns automatically all copyright and therefore the employee can not license his work without the permission of his employer. Consequently, software under a persistent license may actually infringe some third party company's copyright without anyone's consent.

3.4. Rights clearing

The problems with distributed authorship call for *rights clearing*: a company should obtain all rights to the product it wishes to license and make sure there are no hidden liabilities in code contributions from unknown third parties.

Under persistent license, a fully open and distributed development process without sufficient rights clearing is not suitable for any company that wishes to make any direct license sales from their project.

> For example companies distributing Linux such as Red Hat, SuSe, Caldera and Mandrake do not own the copyright to their core products. Because Linux kernel is under GNU GPL and no single entity holds copyright to it they are unable to change the license and make any direct license sales.

To compare, under all permissive license the copyright ownership does not restrict any successive third party from utilizing the software with any means. It is necessary only to make a little modification to the software in order to license it with new terms as a whole.

These copyright transfer problems are further illustrated in figure 1.

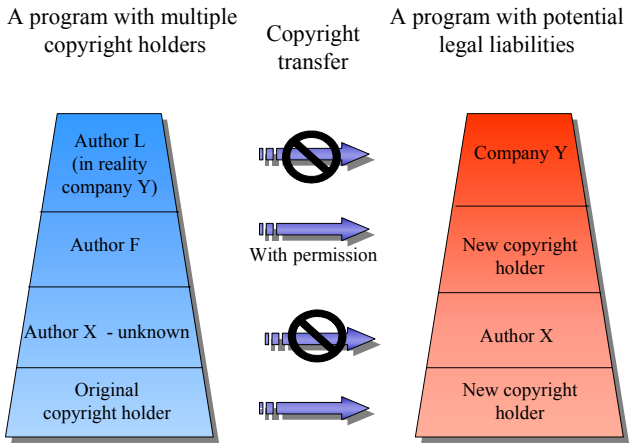

Figure 1. Problems with copyright transfer under distributed development and authorship.

It is possible to think of two ways to clear rights:

- Rewrite the software. This option may mean a lot of work but it is legally the safest bet.
- Obtain rights with a license term or specific contract. This option leaves the possibility of legal risks if the transfer is somehow incomplete for

example because the code contributor has no authorization to give necessary rights

3.5. License combinations

A specific problem with GNU GPL license is that it is incompatible with many other licenses. That is, works under GNU GPL can not be bundled with works under other licenses unless all rights in the other works are waived in favor of GNU GPL (although interpretation of this term is vague [3]). This is especially crucial for developers if they wish to license their software under GNU GPL. It is also crucial for companies that wish to use software under GNU GPL license as embedded software combined with other type of licenses.

> A recent legal case between MySQL AB and Progress Software Corporation illustrates problems with license combinations. Progress combined their own software under commercial license with that of MySQL AB under GNU GPL and sold it under their own license. MySQL argued that because GNU GPL is incompatible with this kind of combination the license sale was illegal. The case is as of July 2002 still not yet settled.[1]

License compatibility problems are not restricted in the combination of restrictive licenses with GNU GPL. In fact many company-specific open source licenses have the same incompatibility problem. Some of these licenses are listed in the table 2 below:

License	IBM public license	Mozilla Public License	Sun Industry Standards Source License	Nokia Open Source License
Type	**All permissive**	**Persistent and viral**	**Persistent and viral**	**Persistent and viral**
Typical use	**Open Sourced programs from IBM**	**Mozilla-project**	**OpenOffice**	**Research projects funded by Nokia**
GNU GPL-compatible	**No**	**No**	**No**	**No**

Table 2. Company specific licenses and GNU GPL.

4. DEVELOPMENT PROCESS

4.1 What has development to do with the license?

It is essential to make difference between the implications of the license choice to development and distribution. Traditionally, it has not been far from the point to call a software license a distribution license. However, permissive terms of open source licenses have enabled large scale collaboration and distributed development between parties unknown to each other. They have enabled a development method based on third party code contributions and extensive peer review.[2][11]

A closer analysis suggests that licenses may give incentives for developers and they may also give possibilities for controlling the project.

4.2. Development incentives

Missing developer incentives may lead to a situation where an open source project either does not have many eyes watching or the quality of the eyes is inferior to proprietary projects. The question is how to motivate developers.

In case of a commercial licensing model, a company has to rely on financial incentives. Under GNU GPL and LGPL this option is rarely available and it would render the use of open source almost meaningless. In any case, most successful open source software companies seem to use a small group of in-house developers, who are paid salary.

To fully benefit from open source development, a company has to find personal incentives for the developers.[2] These are typically either ideological (oppose Microsoft) or practical (a personal need for certain features in the product). As an example of the former, GNU GPL starts with the following sentences:

> "The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users."

4.3. Development control

Permissive licenses allow the development results to be commercialized by any third party without any compensation paid to developers. While this can be seen as a risk to those whose software marketed at end-users, it allows the existence of complementary commercial software market. A company may use a permissive license for a necessary byproduct they have no resources to develop but what increases the market share of their main product.

> For example, id Software Inc. has allowed level editing possibilities for game software thus allowing longer life-time for its otherwise short-lived first person shoot 'em up games.

From the development perspective there are two main possibilities for a software company to control development process [5]:

- Closed source controlled by license
- Open source controlled by leadership. Even if the license would allow competing products and so called forks they are hardly ever successful because of negative network externalities if the leadership is strong.[11]

5. PRODUCT DISTRIBUTION

5.1. Market focus

The market focus of the software should be perhaps the most important factor affecting the license selection.

If a product is aimed at end users, the benefits from choosing an open source license are little if none. Therefore the license terms can be restrictive. The situation is slightly different if the target platform is Linux or some other free operating system where most of the competing products are under open source licenses.

For developers and other third parties the license should be more permissive to maximize the incentives to get third party support for the product.[5]

5.2. Core and complementary products

Free product distribution is not limited to open source licenses. Indeed, many software companies use commercial and restrictive (shareware, freeware) licenses to distribute either limited or full versions of their software for free.

> Microsoft distributed its Explorer browser for free originally to compete against Netscape. As of now, the browser is distributed for free mainly because it ties users to many proprietary standards controlled by Microsoft or its beneficiaries.

Some companies develop add-on software on open source products, which they then sell under all restrictive licenses.

> For example Covalent and theKompany develop in-house and license with all restrictive terms easy-to-use configuration and management tools for many open source core products including Apache web server and MySQL and Postgres databases.

5.3. Dual licensing

A software producer who owns the copyright and other rights that cover the full product is able to license the software according to market demand. Actually many companies who release their software under persistent licenses also sell the same software under commercial license to those who do not want to be bind by the terms of persistent licenses. This is commonly called *dual licensing* and it gives the only option out of a lock-in situation to persistent license for the user.

> For example MySQL AB developing most used open source database MySQL and TrollTech AS developing QT development tools use this licensing strategy. Both of the firms own all copyrights to their software and generate most profit from the sales of commercial licenses. They control the development with leadership.

There may also be substantial *switching costs* when a company or project changes its licensing policy. [11] Costs are generated from e.g. changing accounting principles, signaling the new license to users and converting all documentation and information databases to comply with the new license.

> For example the Mozilla project has created a lengthy and detailed webpage to inform all users of a change in their licensing policy. The change was caused by potential incompatibility of Mozilla's own open source license with GPL and LGPL. [7]

5.4. Antitrust situations

According to Microsoft argumentation GNU GPL license threatens a healthy "software ecology".[8][9] The argument goes that if some essential Internet standard or component is under GPL then every work based on it, including operating system, would be under its terms.

> Microsoft itself has adopted innovations from open source licensing in their shared source policy. It means that Microsoft may distribute the source code of their products to most trusted and important users for peer review.

Thus, a specific legal problem may be if GNU GPL closes licensing markets for some types of software and creates a lock-in situation. In this case, it is possible to think of an antitrust intervention by the state where it would govern that specific software under GNU GPL may be relicensed under either all permissive or commercial license.

> In GNU GPL license, there is actually a term where the contributor may give the copyright of his submission to Free Software Foundation that may later license the software under other terms.

This barrier to entry is of course present with restrictive licenses vice versa but not with all permissive licenses.

6. WHEN DOES OPEN SOURCE LICENSING MAKE SENSE?

It is difficult to give any general suggestion on license use for some particular purpose. Every open source license has its individual implications and judgment of license choice must be made case by case.

The results of this paper may in any case be of some help to companies tackling with the problems of open source licensing. Among others, these questions are relevant when making the license choice:

- *Market focus.* For end users the license terms may be more restrictive but for developer the terms must be rather permissive.
- *Software patenting.* GNU GPL and LGPL licenses are incompatible with software patents.
- *Competition and leadership.* The risk with a permissive license is that if the project has not strong leadership it may be hijacked by a competitor.
- *Third party developers.* GNU GPL is incompatible with most types of commercial add-on products.
- *Rights clearing.* Distributed and open source development process requires rights clearing with costs and benefits.

Finally, we can present an example table of mass market software companies that use different open source licenses in different ways:

Company	MySQL	Netscape	Covalent	SleepyCat
Type of program	**Database**	**Web-browser**	**Web-server**	**Embedded database**
Right owner	**MySQL AB**	**Mozilla/AOL Time Warner**	**Apache foundation**	**SleepyCat Inc**
Target audience	**System administrators**	**End-users**	**System administrators**	**Third party developers**
Licenses	**GPL/ Commercial**	**MPL/GPL/ LGPL**	**Apache-license/ Commercial**	**BSD/ Commercial**
Business-model	**Selling commercial licenses, services**	**Harming Microsoft, selling commercial add-ons**	**Selling commercial add-ons, services**	**Selling commercial licenses, services**

Table 3. Companies using different open source licensing models.

7. ACKNOWLEDGEMENTS

The Finnish National Technology Agency *Tekes* as well as our industrial partners have generously supported our work. Discussions with many open source companies and developers as well as our colleagues at the Helsinki Institute of Information Technology have given significant guidance for this paper.

REFERENCES

[1] S. Costello, Settlement nears in open source GPL suit, NetworkWorldFusion news, available at http://www.nwfusion.com/news/2002/0305settlegpl.html

[2] J. Feller, B. Fitzgerald. *Understanding Open Source Software Development.* Addison-Wesley, 2002.

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

[4] T. Jaeger, A. Metzger. *Open Source Software. Rechtliche Rahmenbindungen der Freien Software.* Verlag C.H. Beck, 2002.

[5] J. Lerner, J. Tirole. "The Scope of Open Source Licensing", 2002, work in progress, presented at the conference on Open Source Software: Economics, Law and Policy, Toulouse, France, June 20-21st 2002, available at http://www.idei.asso.fr/Commun/Conferences/Internet/OSS2002/Papiers/Lerner.pdf

[6] K. Nichols. *Inventing Software. The Rise of "Computer Related" Patents.* Quorum Books, 1998.

[7] Mozilla Relicensing FAQ, version 1.0, available at http://www.mozilla.org/MPL/relicensing-faq.html

[8] Microsoft: Some Questions Every Business Should Ask About the GNU General Public License (GPL), available at http://www.microsoft.com/licensing/downloads/Gpl-faq.doc

[9] C. Mundie. "Security: Source Access and the Software Ecosystem", presented at the conference on Open Source Software: Economics, Law and Policy, Toulouse, France, June 20-21st 2002, available at http://www.idei.asso.fr/Commun/Conferences/Internet/OSS2002/Papiers/Mundie.pdf

[10] Open Source Definition. http://www.opensource.org/docs/definition_plain.php

[11] E. Raymond. *Cathedral & the Bazaar.* Second edition. O'Reilly, 2001.

[12] H. Varian, C. Shapiro. *Information Rules.* Harvard Business School Press, 1999.

[13] S. Willams. *Free as in Freedom. Richard Stallman's Crusade for Free Software.* O'Reilly, 2002. Available at http://www.oreilly.com/openbook/freedom/