IPR CONCERNS AND LEGAL ENFORCEMENT OF OPEN-SOURCE SOFTWARE LICENCES

Mr. Zeeshan Hussain Hashmi[*]

Abstract

Software is the information, in the form of computer programs that makes a computer carry out certain functions. In the initial days of the development of software, software programmes were generally distributed free of cost to increase the number of device users. Eventually, after a period of time, software developed their own proprietary market and closed source softwares have taken the place of free distribution of the softwares then after, a new era of software movement for the promotion of open-source software started and continue to expand till date. Today, most of the companies are actively using or distributing the 'open-source' inside the programme or embedded in the physical product but open-source software does not mean that users can use the software in any way that they might like. Each part of the OSS which we call source code is attached with the licences. Owing to the this discourse the author has formulated a problem for the present area of research that Whether an effective legal protection and enforcement mechanism for protecting open-source software exists in developing countries or only developed countries are reaping out the benefits arising out of commercializing the Open-source Software. The Proposed article will aim to examine relation between Open-source software and protection mechanism available in domestic and international level.

[*] Research Scholar @ University of Rajasthan

INTRODUCTION

Software is the information, in the form of computer programs that makes a computer carry out certain functions.[1] Free and open-source software are ubiquitous now, the terminology can be confusing and mind-numbing and even intimidating, and sometimes you might think it's pointless and just a bunch of magic incantations that you just have to chant in the correct way and hope that everything works out all right. But really, it turns out that the core ideas that underpin the system are actually pretty easy to understand, and if you take the time to do so, this article gives the understanding over the new tools, new ways to influence and control how your software is used, and to help you make sure it achieves the understanding over the Structure of open-source software licences. the objective of this chapter to learn the basics of how intellectual property law in particular, copyright works, to understand how licenses are built on top of those principles and then to enable you to work effectively with free and open-source software both as a consumer of someone else's work and as a creator.[2]

Today, most of the companies are actively using or distributing the 'open-source' inside the programme or embedded in the physical product but open-source software does not mean that users can use the software in any way that they might like. Each part of the OSS which we call source code are attached with the licences viz.

Apache License 2.0

BSD 3-Clause "New" or "Revised" license

BSD 2-Clause "Simplified" or "FreeBSD" license

GNU General Public License (GPL)

GNU Library or "Lesser" General Public License (LGPL)

MIT license[3]

[1] Black's Law Dictionary (10th Ed. 2014), Available at Westlaw BLACKS.
[2] Shemtov Noam & Walden Ian, *Free And Open Source Software, Policy, Law And Practice* 7-30 (Oxford University Press), (2014).
[3] The full details of the Open Source Software Licences, Available at: https://opensource.org/licenses

The users must be aware should understand before using, distributing, changing the software and to make it a derivative work otherwise using, distributing, changing that might lead the act or omission to intellectual property right violation if the use is in conflict with the terms and conditions of the license.[4] So this article will discuss how such open-source software licenses are enforceable and the kinds of intellectual property that will be infringed by noncompliance with the conditions attached with the open-source software licences.

OPEN-SOURCE SOFTWARE LICENSING

It's very normal to be heard that open-source software are risky but very few understand why? Or maybe many of them want to know more about what open-source software is licencing. if you are a programmer of course you already know how much you rely on open-source, no matter who you are you use open-source software every minute, it's in your phone it sends you email and it powers the web. But a lot of things what open-source normally do is behind the scene, so it might not know about it if you are not in the tech business.[5]

These days people like to say that software has eaten the world what they mean is that software now controls almost everything we do, it's not just in the technology that supports your workday but the systems that help you drive, your car, the timer in your oven and even the switch that sets the firmness of your mattress, but now a days people say that open-source software has eaten software because most of that software that powers the computing of the world is open-source now, it means that some programmer wrote the software and then gave it to the world for free.[6] Open-Source Software powers almost all websites and is in almost all Computer Systems but from an Intellectual property perspective it is the opposite of traditionally licensed software.[7]

So, whether you are aware of it or not you are use open-source software every minute of your day. Open-sources is the way of Licensing software but more importantly it's a collaborative way of developing software, the licenses are meant to enable the development while this chapter will

[4] Fitzgerald Brian, *Legal Issues Relating to Free and Open Source Software*,12, 12-29 J.L. & Inf. Sci. (2001).
[5] Hannu Järvinen, *Legal Aspects of Open Source Licensing*, University of Helsinki, Department of Computer Science (2002)
[6] Andrew M. St. Laurent (2004), *Understanding Open Source and Free Software Licensing,* O'Reilly Media, USA.
[7] Dennis M. Kennedy, *A Primer on Open Source Licensing Legal Issues: Copyright, Copyleft, Copyfuture,* 20 St. Louis U. Pub. L. Rev. 345 (2001), Available at: http://www.denniskennedy.com/opensourcedmk.pdf

discuss about the licenses.[8] It's important to remember that the license model serves the development model and not vice-versa. let's look at the development model so we know what open-source is all about the development model is described in the seminal article by Eric Raymond entitled the cathedral and the bazaar proprietary software development is like a cathedral funded by one organization and directed by a master builder open-source software is like the marketplace of ideas anyone can copy or change it and the community decides which version will become the accepted version of the project.

You must have seen the Logo of apache foundation which runs many important open-source projects that are available under permissive licenses the other is the logo of free software foundation that makes software available under copyleft licences like the GPL. These two types of licenses represent two different but related philosophies, free software and open-source software. There are some licences that might have been heard, Open-Source Initiative or OSI is the organization that reviews licenses to certify that they are open-source licenses which means they meet the open-source definition, OSI has approved over 100 licenses but most of them are rarely used, there are basically two categories of open-source software licenses:

1. **Permissive Licences**

The first category is permissive licenses, and these are the simplest kind. They're often very short, very quick to read through, and they almost but not quite but they almost just say that you can do pretty much whatever with the software. One can even build proprietary code on top of the code that's been shared. Usually, it just has to provide attribution, meaning you have to give credit to the person who wrote it, and there's often or usually there's also a warranty disclaimer that says that the author isn't responsible for what you do or how badly you mess up using the code.[9]

You should not use a license that does not have a warranty disclaimer. Now, you would generally pick a permissive license like this when your goal is just for the code to be shared and used as widely as possible. You're trying to make life as easy as possible for developers by making everything available to them for whatever use they want, even if they don't want

[8] Daniel B. Ravicher, *Facilitating Collaborative Software Development: The Enforceability of Mass-Market Public Software Licenses*, 5 Va. J.L. & Tech. 11 (2000)

[9] Anna Haapanen, Free And Open Source Software Licensing and the Mystery of Licensor's Patents, Doctoral dissertation to be presented for public examination, by due permission of the Faculty of Law at the University of Helsinki, in Porthania Hall III on April 1, 2017 at 10 a.m. P-18

to reciprocate by sharing their improvements back to you.[10] That's a very pragmatic approach. It's very business-friendly. And having companies who care about improving the quality of your code and who can pay people to work on it, that can be really powerful.[11] Just be sure that you really would be fine with seeing someone make a lot of money off from your work and give you nothing in return, because that can and does happen, and it's well within their rights. So, people from the communities around these licenses usually talk about the term "open-source," not about "free" software, and it's very non-political, non-activist.[12]

Permissive licenses like the BSD license are very easy to understand they allow you to do anything you like with the software under only one kind of condition which is that if you distribute the software, you must include a copy of the license. this copy of the license serves to inform recipients, that the software they are getting contains some software under this license but it doesn't limit how you can license a product that contains this software, that's why it's called permissive it's a very lightweight requirement and most companies allow use of software under permissive licenses with little or no legal review.

2. Copyleft Licences

The second major family of licenses that I want to talk about are the copyleft ones. Historically this is the category associated with the free software movement, which is the activist and political part of the community. It's also really the origin of most of this movement, These days, maybe some of the revolutionary zeal has fallen away and people are focusing more on the practical benefits of developing open-source software, but it's worth understanding where all this came from and what the creators of these licenses intended. So, the motivation behind a "copyleft license" is to promote user freedom. It's not to make life easy for developers. It is not to let you bootstrap your start up faster. It is not to invite collaboration from industry, although, you know, those entire things can and do happen, and that's great, but the goal is really to guarantee that users to the freedom to study it and change it, the freedom to share your program, and the freedom to redistribute your changes as well. Now to guarantee those freedoms, the central idea that makes a

[10] Joanne van Erp Montague,Davis Wright Tremaine LLP, Copyright Issues with Open Source Software, Prepared for: The 49th Annual Conference on Intellectual Property Law, The Center for American and International Law
[11] Available at: https://www.youtube.com/watch?v=gF4b1TA5Q5w
[12] Legal aspects of free and open source software, Compilation Of Briefing Notes, Workshop Tuesday, 9 July 2013 Jan 4 Q 1, Policy Department: Citizens' Rights And Constitutional Affairs

license a "copyleft" this licence requires you to share the changes you make under the license terms that others shared them with you.

This ensures that even if many people layer changes on top of the code you've released under a copyleft license, the end users of that product will still have those freedoms that you wanted to give them. Copyleft licenses don't prevent you from making commercial software, but they do prevent you from making proprietary software. Now, unlike permissively licensed code, which people can take and do incorporate into closed-source software, copyleft code spreads copyleft to other code that mixes with it. This also means that you can incorporate permissively licensed code into a copyleft project, but not the other way around.[13]

All open-source licenses are unrestricted licenses, they grant all the rights under copyright law. That's one of the requirements of the open-source definition however they impose conditions on the exercise of the right. proprietary licensing like an end-user license withholds some rights open-source licensing doesn't restrict what you can do but if you decide to do certain things you must meet some conditions for copyleft licenses that means you must share changes to the source code.

Copyleft can be boiled down to two conditions this is the hardest concept and open-source licensing once you learn this the rest is easy, if you distribute in binary form, you must make the corresponding source code available to binary recipients and you can only license that source code on the same copyleft licensing terms, this applies to the original copyleft software and any modifications you make to it to create your binaries. The modifications are necessary so the source code corresponds to the binaries, if you distribute in source code form your job is done.[14]

SOFTWARES ATTACHED WITH "NO LICENCE"

After sighted so much of complexities if any developer thinks that this is a mess! I don't know what to do." and you conclude that you don't care about licensing, and what you want to do is just throw the software out in the open platform and let anybody do whatever they want, and want to release his software with "No License". Unfortunately, it doesn't

[13] Gabriella Coleman, Code Is Speech: Legal Tinkering, Expertise, and Protest among Free and Open Source Software Developers, Cultural Anthropology, Vol. 24, No. 3 (Aug., 2009), pp. 420-454

[14] Mark Alexander Zschoch, *A Comparative Analysis of Free and Open Source Software Public Policy in Belgium, Canada, and Germany,* Doctoral Thesis submitted in the Department of Political Science Faculty of Arts and Social Sciences at SIMON FRASER UNIVERSITY Fall 2012

work like that. Just like you can't opt out of other laws that you don't like. So, if you don't have a license, no one has any rights to the code, even if you wish they did and even if you put it up on GitHub. Now, some people still argue, like, "Oh, but it doesn't matter "because I'm not going to sue anybody or do anything like that." "You can use my code." Great. That's really nice of you to say.[15] Please, put that promise in writing and distribute it with your code, preferably in a file named "license." Except don't use your own words, just pick one of the licenses that we talked about earlier.[16]

SOFTWARE IN "PUBLIC DOMAIN"

Now, this point is to be noted that "What about the public domain? Can I just put my stuff "in the public domain and forget about licenses?" And for those who don't recall, the public domain is what we call things that are outside of copyright, either because the copyright has expired, or it wasn't eligible for copyrighted protection in the first place, or it was deliberately set free in this way. And yeah, you will encounter a lot of things out there that are attempting to sidestep the whole issue by more or less placing things in the public domain. Some examples of this are the Unlicensed and Creative Commons Zero. This is much better than nothing, but overall, I still can't really recommend it. One should protect oneself by including the warranty disclaimer language that licenses usually have. Even the Unlicensed website, which is a militant anti-license site, even they suggest that as an option. Unfortunately they hid it away in the frequently asked questions, so it's a bit hard to find, but even they agree that you should have a warranty disclaimer for code. But the reason I can't recommend the public domain approach has to do with those moral rights because these are the rights that talk about right to attribution. And in some places they are irrevocable and forever.[17]

SOFTWARE ATTACHED WITH "SELF WRITTEN TERMS AND CONDITIONS"

[15] Robert L. Greenberg, Open Source Software Development, Thesis for Senior Honors at Brandeis University, May 9, 2003

[16] Ravindra Chingale, "Protection of Computer Software and its Impact on Indian Industry: Revisiting Indian Patent Laws", Doctoral Thesis submitted to National Law University, Delhi, India, Chapter - 1 Introduction, 2, (2015)

[17] Alma Orucevi c-Alagic, "Understanding Software Development in an Open Source Context: Network Analysis of Source Code Repositories", Doctoral Thesis submitted in the Department of Computer Science, Lund University,2016

A Developer should not write its own license terms for software. Because Sometimes people correctly understand that they need a license, but they also think that their situation is somehow really unique and that they have to make up their own new license. Or more likely, they have to give a pile of money to a lawyer who will happily accept it in exchange for doing minimal work by just rephrasing the same stuff that's already in every other license.[18] Because the truth is, there is already a license out there for every need you can think of. Find one that works for you. Every new license that shows up in the world just makes a lot of extra work for developers and lawyers who have to evaluate it and figure out whether it's compatible with all of the other licenses they're using. you know, exponentially, and it imposes this very real cost, and you lose out on the benefits of using common licenses, like shared legal precedence and the common understanding and goodwill that the community around those existing licenses have built up.[19]

COPYRIGHT AND OPEN-SOURCE SOFTWARE LICENSES

Copyright is what this talk is mostly about. The idea is that to encourage people to make expressive, creative works, we have laws that give them exclusive rights to copy, distribute, build upon, and modify that work, even though there may be no technical restriction that would otherwise keep anyone else from doing the same.[20]

Copyright is given to the creator of a work completely automatically. You don't have to do anything to get copyrights or to keep them. In particular, you don't need to stick a little bit of text, like "copyright" or "©". You don't need to put that kind of mark on your work. Anyhow you will get the copyright completely automatically. And that gives you the right to control how it's used for a limited time. Actually, I say "a limited time," and that's the theory, but in practice, no copyrights have actually expired in the United States since the '70s. Every time that we get close to, what would be the new deadline; the law gets changed retroactively to extend the duration. Moral rights are also a very closely related concept to copyright. In some countries, they're considered just a subset of copyright, and in some places, they're a little bit different, and in some places, they don't even exist at all.[21]

[18] Available at : https://www.youtube.com/watch?v=9kGrKBOytYM
[19] Ibid
[20] Joanne van Erp Montague,Davis Wright Tremaine LLP, Copyright Issues with Open Source Software, Prepared for: The 49th Annual Conference on Intellectual Property Law, The Centre for American and International Law.
[21] Brian Fitzgerald; Graham Bassett; Larry Rosen; Bill Lard, Legal Issues Relating to Free and Open-Source Software, 12 J.L. & Inf. Sci. 159 (2001)

Moral rights are sort of the non-commercial part of copyright. It's not so much about making money off from your creations.[22] It's about things like the right to be identified as the author, or the right to the integrity of the work. And "integrity" in this context means keeping people from changing or presenting your work in ways that you don't like, like putting on a display and defacing your art, even if they own it. Moral rights also behave differently. For example, they might not be time-limited. They might actually last forever. They might even last beyond your lifetime and pass down to your heirs and descendants. So, your grandkids maybe will have "a say" in what people are allowed to do with your software and in some countries, you cannot waive, reject, or give away or sell your moral rights. It's just not allowed, like how you are not allowed to agree to be murdered.

PATENT AND OPEN-SOURCE SOFTWARE LICENCES

The next big kinds of intellectual property I want to talk about briefly are patents. Where copyright was about creative works, patents are about protecting functional inventions. The deal is that you share the details of how your new gizmo works, and society in exchange gives you this exclusive monopoly on the invention for a little while.

The owner of a patent might try to get their patented software technique adopted as part of a standard, so that anyone who implements that standard then has to pay them money for a license or risk being sued Or they might even actually themselves contribute the code that implements their patented technique, and then later, either because they planned this in advance or just because they changed their minds, they might come back later and try to sue you for using that code that they themselves contributed to your project. So, some licenses have protections against this kind of thing, and when we talk about those licenses. I will call that out specifically, because it is a great thing to have in a license. Also, as a side note, there's something called a design patent, which covers the way a product looks, and that is not the same thing as a regular patent. It doesn't work the same way. It's not meant to cover the same kind of things but because it also has "patent" in the name.

TRADEMARKS AND OPEN-SOURCE SOFTWARE LICENCES

The next type of intellectual property is trademarks. The idea behind a trademark is to protect consumers from fakes. That might be trying to take advantage of the good reputation of a particular brand by imitating or copying their name or logo or identifiers like

[22] Ibid

that.[23] Now this doesn't usually come up in software licenses directly, but it can definitely be an issue for software projects, especially user-facing ones that have brands and reputations to protect. You might remember that in 2013, the Python Software Foundation actually had to fight for the name "Python," the Debian GNU/Linux distribution had to ship Firefox under the name "Iceweasel" with a really cute different little blue logo because of disagreements about trademarks, even though the licensing of that code was perfectly fine.[24] That was a friendly disagreement, though, and both sides had very good points, and they've now sorted it all out, which is great. But again, most licenses don't have anything to say about trademarks, but you should know what they are because they do come up for software projects. If you expect end users to be able to recognize the name and brand of your project, then maybe spend some time to think about what rights or requirements you might want to give to people who change your code, can they associate that new project with your brand? In general, though, just try not to rip off anyone's name or logo.[25]

CONCLUSION

There are other kinds of intellectual property as well, but they're not really relevant to software licensing, so we're just going to ignore them. That works out well. So, why do we need licenses at all? Well, intellectual property law is a reality whether you like it or not, and since copyright is both so powerful and automatic, we have to live with it and operate within its boundaries. But it's not just a bunch of restrictions. We can actually also work within the confines of the law to achieve the outcome that we want for our software. We can use the laws in ways that are beneficial and that actually help spread our work. So, a license is a set of permissions that you give to someone. When you create something like a piece of software, you alone have the right to it, and you have control over how to share those rights with others. And a license is what we use to give people rights they wouldn't ordinarily have, like the right to use the software, to modify it, or to copy and share it. Licenses can also be used to set conditions or create obligations.

[23] T. Dare and H. Anderson, *Passport Without A Visa: Open Source Software Licensing and Trademarks,* International Free and Open Source Software Law Review, Volume I, Issue 2, 2009, p. 102

[24] R.M. Stallman, *Why Free Software is better than Open Source*, in Free Software, Free Society: The selected Essays of Richard M. Stallman, Free Software Foundation Inc., 2002, at 57 https://www.gnu.org/philosophy/free-software-for-freedom.en.html.

[25] Y. Welinder and S.LA Porte, *Hacking trademark law for collaborative communities*, Hacking Trademark Law for Collaborative Communities, in SSRN, 2014, at 18

The most obvious example is of course proprietary software, which generally has a license that only gives you the right to use it, but prohibits you not just from copying and sharing it, but also from things like reverse engineering it, or in some cases even from really wild stuff like benchmarking the performance of the software and sharing that data. Some proprietary database licenses have that. And I mean, that's a pretty odious example of a requirement in a proprietary license, but free and open-source software licenses usually also have obligations and requirements that come with them, they're just of a different nature, and we'll look at some of those in a minute. Usually if you don't comply with the obligations that a license creates, you lose all the rights that it would otherwise have given you, and there might be other penalties on top. But the key idea is that without a license, no one has any rights to the software, and the license is the thing whereby the creator grants those rights. If you are just publishing the source code, like by throwing it up on Github without a license, that does not give anyone any rights. That is not code you can use, it's just code you happen to be able to look at, like how you can probably see my laptop sitting up here right now and you can look at it, but you don't have the right to take it home.

So now, we've covered the core ideas of intellectual property law, and we've covered sharing the code, accepting contributions from others, practical things to keep in mind, and that is pretty much what you need to know about free and open-source software licensing. So, to protect everyone from the violation of open-source software, end user must pick a license that's already established, proven, and in wide use. The Open-Source Initiative, the Free Software Foundation, choosealicense.com they all maintain really good online resources with lists of licenses.