SI 413, Unit 10: Software Licenses

Daniel S. Roche (roche@usna.edu)

Fall 2023

1 Software Uses and Permissions

Whoever writes a piece of code *owns* it; only they have the power to compile, run, and share that code. Often, they want to grant some other people permissions as well. That's where software licenses come in.

Specifically, the author or owner of a piece of software might want to let others do the following with it:

- **Use**: This is the most basic right. It says that you can run the compiled software on your own machine, as often as you wish.
- **Distribute**: You can make copies of the software and hand them out to others, which nowadays usually means put it on your website for others to download.
- **Inspect**: The *source code* of the software is available to be perused. This means that someone can tell how the software actually works.
- **Modify**: Said source code can be changed to make a new or related piece of software, or it can be used as a subcomponent in another piece of software.
- **Redistribute**: Any modifications can be independently published or handed out.
- **Sublicence**: The terms of the original licence can be changed by any redistributor. Notably, you can make a modification, and then sell it, even if the original licence specified that the software was free.

There are more "rights" or "privileges" that might be granted with a piece of software, but this is a good start. Of course you know that not every piece of software comes with all these rights! Most don't, in fact.

And even if they do, there might be certain "restrictions" attached, such as:

- **Payment**: You can only use the software if you pay money for it.
- **Reverse engineering**: Any user promises not to even try to inspect it. Yes, this is a restriction that exists on many pieces of commercial software!
- **Attribution**: If you modify and redistribute the software, you have to give credit to the original authors
- **Copyleft**: If you modify and redistribute the software, you must license your software with the same license as the original. To point, if a piece of free, open-source software has a "copyleft" restriction, then it can't be used as part of a commercial, closed-source program that is sold for profit.

Wow, that's a lot of different options when it comes to software! And this is just a rough summary - there are many more fine details that can (and have been!) considered when it comes to releasing software into the world.

2 Licence categories

If the options seem overwhelming when considering your next software project that you want to publish, don't dismay. Many have trod this road before you! Here are some of the most common kinds of licences that get used nowadays.

2.1 Closed-source

In the first category are *closed-source software programs*. Closed-source just means that the source code is kept secret, only for the original developers or owners to see. Here are the common licencing options.

- Paid-for programs like Microsoft Office and AutoCAD fall under the category of *proprietary*, or *commercial* software licenses. They generally have minimal permissions, and maximal restrictions. Basically, you can use the software, but not distribute or inspect it, and only then if you yourself have paid for it.
- A slight lessening is seen in *shareware* and *freeware* programs such as WinZip or ITunes. You can download full or partial versions of these programs for free online. Sometimes you can even turn around and redistribute them on your own website if you want. But the source code is still not made available, so inspection, modification, and the like are still not possible.

2.2 Open-source

The second category, as you might guess, are *open-source* software programs. This is also called "free software" (which doesn't just mean that you don't have to pay for it). These developers have decided to publish the source code along with their program for anyone to see. But there are still a number of choices to be made. Here are the most common two options:

- GPL (Gnu Public Licence) is the free software license used by the GNU/Linux operating system, and championed by the Free Software Foundation. This is the open-source license for the activist: it is completely free software, that anyone can use, inspect, modify, and redistribute, but it comes with the *copyleft* restriction. That means that you can't use GPL software in your own program, then turn around and make it closed source. The original goal was to encourage more software to be released open-source; in practice, another consequence is that many companies don't want to use GLP or copyleft-licensed code.
- Permissive licenses is the general term for open-source licenses that don't have the copyleft restriction. Sometimes they still require *attribution* (i.e., "give me credit"), but usually not much else. The most famous example is the BSD (Berkley Standard Distribution) license that was created for the first free version of the Unix operating system. This license is more about getting your code out there, not about spreading the cause of free software. It can be used by and even just repackaged in a commercial, closed-source, must-pay version.

You should definitely look at some web sites like the Open Source Initiative and the Free Software Foundation and check out all the free, open-source licenses there are. There are many options to make your code free, depending on what you want for yourself and for the community of open-source developers.

3 Case studies

There are many thorny legal issues involved with all this. The way software licensing works in theory and in practice is constantly evolving as the technology itself advances.

Here are some recent (and non-so-recent) articles about recent legal battles over the definition of open-source and software licensing:

- John Deere tractors use Linux and other copyleft-covered software, but may be violating the GPL licensing terms by refusing to release the source code for the software running on their equipment:

  https://www.theregister.com/2023/03/17/john_deere_sfc_gpl/

- Hashicorp decides to move from "open-source" to "source-available" with a license that restricts how it can be used, particularly in the cloud

  https://thenewstack.io/hashicorp-abandons-open-source-for-business-source-license/

- A major lawsuit is brewing against GitHub (and by extension, Microsoft and OpenAI) for sucking up all kinds of source code and incorporating it into proprietary AI tools

  https://www.theregister.com/2023/06/23/open_source_licenses_ai/

- The Red Hat software company (now owned by IBM) is restricting access to its source code, while still claiming that it "uses and will always use an open source development model":

  https://www.itworldcanada.com/article/red-hat-decision-turns-world-of-open-source-linux-upside-down/543157

- A court rules against TV manufacturer Vizio that software licenses (and in particular, copyleft restrictions) do have the force of a contract and have to be followed:

  https://www.theregister.com/2022/05/16/vizio_gpl_contract/

- Amazon's Kindle e-book is based on Linux (licenced under the copyleft GPL license), and yet the Kindle source code itself is not really open-source:

  http://arstechnica.com/information-technology/2009/06/amazon-code-release-irrelevant-kindle-is-still-closed/

- The newest version of the GPL license won't be applied to the Linux kernel:

  http://news.cnet.com/torvalds-no-gpl-3-for-linux/2100-7344_3-6031504.html

- A landmark court case that determined software vendors can restrict your ability to resell software, unlike if you purchase a CD or a book:

  https://www.eff.org/cases/vernor-v-autodesk