Intro to Methods of Software Engineering Fall 2018

Lecture 12 — November 6, 2018

Patrick Lam *version 1*

Software Licensing

Recall our discussion about intellectual property. Let's talk about software in more detail.

How does IP apply to software? Copyright and patents.

Need to have permission of the copyright owner to run the software on your computer. We won't talk about patents at all today, but software that violates patents may be not legally usable.

Getting software. These days, perhaps the main way of getting software is by downloading it. Often you don't have to pay money to download the software, but you may need to do an in-app purchase or otherwise acquire a key.

We'll talk about revenue models next week. But many software companies find it much easier to stay in business by "renting" rather than "selling" software, e.g. Adobe.

Your rights with random downloaded software. Presumably if you download it from the copyright owner, you have the right to execute the software. As I mentioned above, maybe you have the right to execute it for a limited time. By default, we consider software to be under a *proprietary* license.

This software is built from source code, just like you are creating in CS 137 and like you created with the Spaceship project. But you usually don't get the source code when you download the software. That makes it hard to remix the software.

What if you have the source code? Either you have extracts from the source code, in which case you probably can't run the system, or you have the complete source code in buildable form.

If you have the complete source code, you would have the ability to build and modify the software. But you may not have permission to do so, depending on the license.

For instance, as of April 2017, the owners of Unix Research Editions 8, 9, and 10 granted permission to use and modify the software in non-commercial contexts[1]. So, Microsoft could not use that software in building its products. But you can tinker with it for academic purposes.

Free software licenses. I briefly mentioned that Free Software hacks the copyright system in pursuit of the goal of freedom. The GNU General Public License is the most common free software

[1] `https://www.theregister.co.uk/2017/03/30/old_unix_source_code_opened_for_study/`

license. It grants permission to use, modify, and redistribute software (including remixing), as long as the derivative works remain under the same license. Another way to think about it is "share and share alike". The authors welcome you to use their software, but you can only use it in your own product if your product also is shared with the world.

Permissive licenses. Some software authors grant even more rights to users of their software. The MIT license is one example of a permissive license. Under the MIT license, anyone can use, modify, and redistribute the software, as long as they agree that there is no warranty and as long as they preserve the authorship notices.

Supporters of Free Software worry that software under permissive licenses can be incorporated into proprietary software. This is appropriate, for instance, when software authors want their work to be used by as many people as possible. However, it does not expand the universe of free software.

Case Study. The lerna project had a kerfuffle in August 2018 about licenses. This project is distributed under the MIT License. Someone sent in a pull request suggesting that certain corporations be banned from using lerna based on their alleged support for US Immigration and Customs Enforcement.

`https://github.com/lerna/lerna/pull/1633`

After discussion, the proposed change was not accepted into the project. (Had it been accepted, banned users could also have validly used earlier versions; this is known as a *fork*.)

What are the ethical considerations involved in this license change? Are you in favour of it or not? What are the broader consequences of the change?