

Releasing Free Software If You Work at a University

GNU philosophy

This essay was originally published on <http://gnu.org>, in 2002.

This document is part of GNU philosophy, the GNU Project's exhaustive collection of articles and essays about free software and related matters.

Copyright © 2002 Richard Stallman

Verbatim copying and distribution of this entire document are permitted worldwide, without royalty, in any medium, provided this notice is preserved.

Releasing Free Software If You Work at a University

In the free software movement, we believe computer users should have the freedom to change and redistribute the software that they use. The “free” in “free software” refers to freedom: it means users have the freedom to run, modify and redistribute the software. Free software contributes to human knowledge, while nonfree software does not. Universities should therefore encourage free software for the sake of advancing human knowledge, just as they should encourage scientists and other scholars to publish their work.

Alas, many university administrators have a grasping attitude towards software (and towards science); they see programs as opportunities for income, not as opportunities to contribute to human knowledge. Free software developers have been coping with this tendency for almost 20 years.

When I started developing the GNU operating system, in 1984, my first step was to quit my job at MIT. I did this specifically so that the MIT licensing office would be unable to interfere with releasing GNU as free software. I had planned an approach for licensing the programs in GNU that would ensure that all modified versions must be free software as well—an approach that developed into the GNU General Public License (GNU GPL)—and I did not want to have to beg the MIT administration to let me use it.

Over the years, university affiliates have often come to the Free Software Foundation for advice on how to cope with administrators who see software only as something to sell. One good method, applicable even for specifically funded projects, is to base your work on an existing program that was released under the GNU GPL. Then you can tell the administrators, “We’re not allowed to release the modified version except under the GNU GPL—any other way would be copyright infringement.” After the dollar signs fade from their eyes, they will usually consent to releasing it as free software.

You can also ask your funding sponsor for help. When a group at NYU developed the GNU Ada Compiler, with funding from the US Air Force, the contract explicitly called for donating the resulting code to the Free Software Foundation. Work out the arrangement with the sponsor first, then politely show the university administration that it is not open to renegotiation. They would rather have a contract to develop free software than no contract at all, so they will most likely go along.

Whatever you do, raise the issue early—well before the program is half finished. At this point, the university still needs you, so you can play hardball: tell the administration you will finish the program, make it usable, if they agree in writing to make it free software (and agree to your choice of free software license). Otherwise you will work on it only enough to write a paper about it, and never make a version good enough to release. When the administrators know their choice is to have a free software package that brings credit to the university or nothing at all, they will usually choose the former.

Not all universities have grasping policies. The University of Texas has a policy that makes it easy to release software developed there as free software under the GNU General Public License. Univates, in Brazil, and the International Institute of Information Technology in Hyderabad, India, both have policies in favor of releasing software under the GPL. By developing faculty support first, you may be able to institute such a policy at your university. Present the issue as one of principle: does the university have a mission to advance human knowledge, or is its sole purpose to perpetuate itself?

Whatever approach you use, it helps to approach the issue with determination and based on an ethical perspective, as we do in the free software movement. To treat the public ethically, the software should be free—as in freedom—for the whole public.

Many developers of free software profess narrowly practical reasons for doing so: they advocate allowing others to share and change software as an expedient for making software powerful and reliable. If those values motivate you to develop free software, well and good, and thank you for your contribution. But those values do not give you a good footing to stand firm when university administrators pressure or tempt you to make the program nonfree.

For instance, they may argue that “We could make it even more powerful and reliable with all the money we can get.” This claim may or may not come true in the end, but it is hard to disprove in advance. They may suggest a license to offer copies “free of charge, for academic use only,” which would tell the general public they don’t deserve freedom, and argue that this will obtain the cooperation of academia, which is all (they say) you need.

If you start from values of convenience alone, it is hard to make a good case for rejecting these dead-end proposals, but you can do it easily if you base your stand on ethical and political values. What good is it to make a program powerful and reliable at the expense of users’ freedom? Shouldn’t freedom apply outside academia as well as within it? The answers are obvious if freedom and community are among your goals. Free software respects the users’ freedom, while nonfree software negates it.

Nothing strengthens your resolve like knowing that the community’s freedom depends, in one instance, on you.