Free software and research: a rationale and a tutorial

Francesco Potortì
ISTI - CNR
via Moruzzi, 1 – I-56124 Pisa
Email: Potorti@isti.cnr.it

***Abstract*— "Free software licenses are a natural choice in a research environment." In the following, we will try to back this simple statement with some considerations and examples, in an effort to analyse the significant interactions between free software and research. An appendix lists and describes the most common terms used wben speaking about software licenses and suggests guidelines for choosing a free license.**

I. What is free software

Free software is rooted in the concepts of freedom of speech and free exchange of information. In scientific environments, the latter concept is especially prized. Free and easy diffusion of information is generally regarded as one of the main forces behind the exceptionally fast growth of scientific knowledge over the last three centuries.

However, the freedom of exchanging ideas is not simply a practical matter: it lies at the base of the concepts of freedom of thought and freedom of expression. Just like ideas, software is immaterial, and can be easily reproduced and transmitted. Just like ideas, its growth and its evolution is advantaged from free diffusion. And just like ideas, more and more software is involved in society, producing effects that are ethical, economical, political and in a general sense, cultural.

During the Eighties, Richard Stallman formalised the concept of free software for the first time. Stallman's definition is widely recognised as the canonical definition of free software. It consists of four rules, the *four freedoms*:

Freedom 0, or fundamental Freedom: The freedom to execute the program, for any purpose.

Freedom 1: The freedom to study the program, and adapt it to your needs.

Freedom 2: The freedom to redistribute copies.

Freedom 3: The freedom to improve the program, and release your improvements to the public.

Freedoms 1 and 3 require access to the source code. *A program released with a software license that grants the four freedoms is said to be free software*. Notice that free in *free software* refers to freedom, not price.

II. Software licenses

A software license is a legal document that accompanies a program. Without a software license, according to the provisions laid down within the Berne copyright convention, *a program cannot be copied or modified without the explicit permission of the authors*. A free software license, on the other hand, allows you to study, copy, modify and release the modified program.

Most of the commonly used software licenses are *proprietary licenses*, i.e., they do not grant the four freedoms. They do not usually allow users to freely copy or modify the program. Often they do not even allow you to keep separate copies of the program on your desktop and laptop computers, to make a trial copy on a colleague's computer, or to keep a complete backup installation in case your main computer crashes. Since proprietary programs are typically distributed without the source code, they do not allow you to study how they work, or to improve them or adapt them to your needs.

Free licenses, on the other hand, do grant the four freedoms. Free licenses may be copyleft or non-copyleft.

A *copyleft license* is a free license that uses the copyright laws to do the opposite of what proprietary licenses usually do. While proprietary licenses prevent you from copying or modifying the program, *a copyleft license is a free license that uses copyright laws to keep the program free*. This can be explained by saying that the license is *persistent*. If you get a copylefted program and you want to redistribute it, either in modified or unmodified form, you must give the recipients the same rights you acquired when you received the program. A prerequisite for this is that you distribute the source code along with the program. The most widespread copyleft license is the GNU GPL.

A *non-copyleft license* is a non-persistent free license. If you get a non-copylefted program and you want to redistribute it, you are not bound to give the recipients the same rights you acquired when you received the program. Most non-copyleft free licenses allow you to make proprietary versions of the program, or even to just redistribute it verbatim using a different license, even a proprietary one. The most widespread non-copyleft license is the BSD license.

III. Common misconceptions and false assessments

While the keywords *free software* and *open source* have gained wide popularity, they are often misrepresented, giving rise to many false beliefs, even in the specialised press and among technical experts. Here we will try to highlight and correct just a few of these.

A. Open source software is not the same thing as free software

From a practical standpoint Open Source and Free Software are one and the same thing. In fact, apart from some minor licenses, the *open source licenses* identified by the Open Source Definition are the same as the *free software licenses* identified by the Free Software definition.

The difference arises from the principles that gave rise to the two definitions: while free software is defined on the basis of ethical and political reasons, open source is defined on the basis of practicalities and convenience. In this document, open source and free software are used interchangeably, but the latter is preferred because its definition is clearer and more concrete.

B. Open source is a software development method

Strictly speaking, open source is a licensing model, not a development method. Many open source advocates claim that the two are linked, but this remains an opinion, and as such is debatable. Eric Raymond's famous essay The cathedral and the bazaar tries to draw a clear line between centralised and distributed development methods. The former is used for Emacs, the main editor of the GNU project, while the latter is used for the Linux kernel. Raymond's conclusion is that the distributed development method is inherently superior as far as big projects are concerned, and that if you want to use distributed development with a large scale project, you need an open source license.

However, the license and the development method used are largely independent issues, and they should not be confused. In fact, while it is true that distributed development methods can naturally be applied to open source projects, they are also used in the big software firms for developing proprietary programs. On the other hand, since centralised development is almost always easier to manage, it is widely used for open source projects, and is no stranger to proprietary programs.

C. Free software is technically superior / inferior

Both statements are false, in general: there is no established relationship between the license used and the quality of the software. Free software has the potential for greater reliability because anyone can study the source code, and all bugs are shallow to a million eyes, but this potentialities not necessarily translate into reality. The only case where a strong point can be made about the technical superiority of free software is when dealing with security software, such as cryptography or signature programs.

D. Free software costs more / less

Neither of these statements in generally true, because, in most cases, the cost of software depends only partially on the license fees. Installation, training, maintenance, upgrading and customisation typically constitute a much greater portion of software costs, and there is no direct relationship between the license used and the expense needed. It is easy to name cases where free software can cost much more or much less than comparable proprietary solutions.

E. Free software has no copyright / copyleft means non copyright

Neither is ever true, because anything published without a copyright license is subject to the Berne international copyright convention; this means that it cannot be copied or modified without explicit consent from the author. Programs published according to these terms are not free. Free software must be distributed along with a software license that releases the standard copyright restrictions. A copyleft license is a kind of free software license, and is founded on the copyright law.

F. Writing or modifying free software means I have to release it publicly

No free software license forces you to do this. If you write free software for release to a restricted number of recipients, neither you nor the recipients are forced to release it further, whether modified or not. However, you cannot prevent the recipients from releasing the software, if they so wish.

G. Releasing a free program will create a community around it

Creating a community around a program requires much more than releasing it publicly as free software. On the one hand you are not obliged to manage a community every time you release a free program; on the other if you want to create a community you must build and feed it: this is usually a difficult and time-consuming task, though it may be well worth the effort. Note than you can build a community around a proprietary program, but this is generally much more difficult.

IV. Free software and research

Modern science is connatural with the free exchange of knowledge.

All scientific research today relies heavily on free and flowing exchange of information, in all possible forms: congresses, conferences, magazines, web sites, professor invitations, seminars, remote and face-to-face cooperation, common research projects are all considered essential features of the modern scientific environment. They enable cross-fertilisation of ideas, open the minds of researchers, contribute significantly to the birth of new concepts, and form the foundations for the incremental improvement of results. Moreover, these features create a peer-to-peer network of mutual control that makes advancement of science outstandingly reliable – yet efficient – among the complex processes created by human civilisation.

Free software is a natural product of a research environment. The birth and development process of software has much in common with that of scientific ideas. Just like scientific research, software is improved by learning from others' results, a process which is much more efficient if the software source code is disclosed, similarly to disclosing the details of scientific findings.

Growth through the accumulation of results is common to scientific research and software. Isaac Newton said that if he had seen further, it was by standing on the shoulders of giants. Software development shows a similar pattern: most successful programs grow with time, they evolve and improve incrementally. Both in the research and software fields, open knowledge greatly helps the process, which is only possible if modification is allowed.

Science is credible because in principle everyone can check its results. In order to make this principle applicable, researchers are encouraged to publish their results in a form that allows a complete and accurate scrutiny by any independent third party, usually in the form of scientific papers where all the relevant points are detailed. This is similar to the way free software programs can gain credibility: by making their source code available, open to scrutiny by any third party.

Credibility and reliability come hand in hand. Scientific results are reliable because they are independently repeatable. A good scientific paper makes it possible to reproduce the results of an experiment, be it physical or conceptual, by disclosing enough details for independent researchers to reproduce the experiment and verify that the results are the same. Something similar happens for software, where the reliability of a program can be tested by making the source code available for inspection and recompilation on different machines and architectures.

With scientific research, cooperation is the name of the game. Research languishes without cooperation: it is a mental habit for researchers, who should and generally do find it natural to exchange ideas and results. A cooperative environment is as fertile for software development as it is for research: software developers find it easy and natural to exchange pieces of code and ideas, and can benefit from the work of others.

V. Advantages in academia

The production of free software can bring advantages in academia and generally in research environments, from a purely scientific production viewpoint in terms of published papers, from an image standpoint and from a business application perspective in terms of development.

Scientific research works involving or including free software implementations are naturally suited for publication, because:

- Working methods are fully disclosed, because the source code is available. This is something a peer reviewer will appreciate.
- Dissemination of results is encouraged, because the source code can be freely copied and republished. Being cited often increases the value of a paper.
- Results can be easily reused and incrementally improved, thanks to the free software license. This increases the chance for a paper to became a seminal work. It is worth noting that, if the license is copyleft, the code is guaranteed to remain free when improvements are published.

Sporting a corpus of developed and published free software is convenient for research institutions because:

- It represents a source of pride, because it is a showcase where the scientific institution can clearly exhibit its contribution to advances in scientific knowledge and benefits for the public at large.
- It helps to receive contributions from many sources, because researchers are often keen to contribute improvements and corrections to published free software, especially if they use it or are working on it.
- Making it clear from the beginning of a project that it will be publicly distributed with a free software license usually reduces the problems related to copyright attribution. For example, anyone can modify a program previously started by an author who then happens to give it up. Also, authors will usually more willingly contribute to a free software project rather than one that will remain the closed property of an institution.

It is a common belief that, when going commercial, proprietary software is the safest option. However, free software offers many advantages with regard to the relationship between academia and business:

- If, after in-house research, development is handed over to a spinoff, there will be no licensing problems, because no copyright issues will arise between the mother institution and the spinoff.
- Releasing a program as free software is a good way to create and promote new standards. A free software program can be released as a proof of concept, as a reference implementation of the proposed standard, or even as a high-quality, completely usable implementation of the standard.
- Technological transfer is simplified, because no particular copyright issues arise when transferring the programs from the research institution to businesses.
- Spinoffs or internal business sections can coexist with internal research development and mutually benefit from each other, exchanging code in both directions without any particular copyright agreements, as long as the code is exchanged with a free software license.

VI. Encouraging the production of free software in research institutions

We argue that research institution involved with writing software should encourage researchers to use a free software license when releasing software written as part of their research.

In particular, publicly funded research should, as a general rule, disseminate the results produced in software form by publishing them with a free license, so that they can be freely studied, copied and modified. Such a policy would have a positive effect on some of the main objectives of publicly funded research by:

- contributing to public knowledge;
- boosting research advancement;
- creating a software base for the industry to exploit.

There are several ways to encourage free software production in scientific research institutions.

First and foremost, research project financing should normally require that the results of the research be published with a free software license. If no requirement is possible, the financing institution should clearly point out that the aim of the project is the advancement of public knowledge, and consequently the use of free software licenses is encouraged.

In addition to project financing institutions, *all* scientific institutions should have a policy that software produced with public fundings should remain free. Whether the license should be persistent or not (copyleft or not) is a matter of discussion. Generally speaking, the advantage of a non-copyleft license is that it does not prevent embedding free software in a proprietary program, which makes things easier for the software industry, while the advantage of a copyleft license is that it forces to release improvements as free software, thus enlarging the base of available free software.

When a research institution has a policy of financing and encouraging spinoffs, those who base their business on free software should be favoured with respect to the others, because business based on free software creates a healthy software business environment and promotes locally-based work.

From a purely academic point of view, writing and publishing free software should be regarded as a research achievement per se, analogous to publishing a research work by means of a paper on a journal. In order to make this possible, a network of software peer reviewing should be created, similarly to the current procedure adapted with the publishing of scientific papers. We advocate the setup of such a network, and we claim that published free research software should be peer reviewed in the same way that research articles are reviewed, and consequently should be similarly considered a research achievement.

Just as diffusing one's findings is highly prized in research community, diffusing software should also. Research institutions should encourage this cultural process, by openly stating that publishing free software is as important as publishing other research results, and by using internal evaluation processes in accordance.

Just as authors and readers of scientific papers recognise the great value of disclosing the exact procedures described, they should recognise the need for reading and modifying the source of software described in such documents. This should be considered by publishers, editors and reviewers of scientific journals when evaluating a contribution.

APPENDIX
SOFTWARE LICENSES

A software license is a legal document that accompanies a program. Without a software license, according to the provisions laid down within the Berne copyright convention, *a program cannot be distributed or modified without the explicit permission of the authors*. A free software license, on the other hand, allows you to study, redistribute, modify and release the modified program.

A. Proprietary licenses

Any license that is not free is said to be *proprietary*. Proprietary licenses come in a variety of flavours. Here is a brief description of some of the terms most commonly used when distributing software in research environments with a proprietary license.

1) No license : Even if this is not a license, the final effect is that of a proprietary license. Every program that is not accompanied by a copyright license is subject to the Berne international copyright convention, and can not be distributed or modified without the explicit consent of the copyright holders. This means that the program is not free without a free copyright license, even when the source is available, with or without charge.

2) Freeware : Without any other specifications, a program is said to be *freeware* if it can be copied freely at no charge, for any use. Usually does not come with source, or has some other restriction that makes it not free.

3) Shareware : A *shareware* program can be freely copied and distributed, but cannot be rightfully used without paying for it after an initial period of test usage.

4) Restricted use : A common constraint that makes a license non-free is the *non-commercial* clause. Many programs are distributed as *free for non-commercial use*, that is, they can be used without charge as long as no money is gained from ther use. Similar clauses are *for personal use*, *for academic use*, *for educational use*. Other restrictions include the *non trasferable* clause, wich prevents free redistribution. Such programs may come with or without source, usually without.

5) No source available : Most proprietary programs are distributed in binary-only form. The binary may or may not have limits on its usage.

6) Non disclosure agreement : Sometimes programs are distributed only after the recipient has signed a contract (the *agreement*) where the signer agrees not to divulge some kind of information about the program. This is common when the source is distributed, but cannot be further redistributed. For example, the signer can agree not to divulge information about the details of how the program works.

B. Choosing and using a free software license

A free software license is one that allows the recipient of the program to use it for any purpose, copy, modify, and redistribute it. Whether money is charged for distribution has no relevance.

In order to make a program free, you should accompany the source code with a file (usually named `LICENSE.TXT`) containing the text of the license you choose. Moreover, it is very advisable to add a copyright line at the top of each source file, similar to the one that is found at the bottom of this document. When you modify the file, make sure that you add the current year to the list of years in the copyright line. If writing a © character is a problem, use the three characters (C) instead. After the copyright line, write one more line saying that the copyright licensing terms are contained in the accompanying file `LICENSE.TXT`.

Do not try to write your own license. It is very difficult to come up with something well written, even if you are a lawyer.

Also, using an established license has many advantages: people know it and what are their implications, they are written by lawyers, they have long been tested on the field, they make it easier to share code between free software projects.

What license should you choose for your program? It depends on your aims. As also mentioned in the Debian Free Software Guidelines FAQ and an article by David A. Wheeler, we suggest using one of the three following licenses.

1) The GNU General Public License : If you want to be sure that every copy of your program, even modified copies, will be accompanied by the corresponding source, or else by an easy way of getting the source, you should use the *GNU GPL*. This is the most successful license of all, used for the programs developed by the GNU Project and many others. Examples include the Linux kernel, the GNU Emacs editor, the portable GCC compiler, the KDE and Gnome desktops for X.

2) The GNU Lesser General Public License : If you want to be sure that your software remains free even after modification, but you want to leave people free to link it against any program, be it free or proprietary, you should use the GNU LGPL. This is a GPL compatible license, used mostly for libraries, plugins and components. Some notable available under the LGPL include the i386 emulator Bochs, many of the Gnome desktop libraries, the C library Glibc, and the C++ library libg++ .

3) The BSD-like licenses : If you aim at maximum diffusion for your code by allowing its use in any program, be it free or proprietary, you should use the new BSD license (also called the BSD license without the advertising clause) or the MIT X license. Successful programs released under one of these licenses include the kernel and core utilities of the FreeBSD operating system, the XFree86 X server, many networking utilities, and the Apache web server.

Acknowledgements

I'd like to thank Prof. Piero Maestrini for his continued support and his ideas about advocating peer reviewing of software. I also thank Associazione software libero, of which I am a member, for discussions and encouragement.

Copyright