

FREE SOFTWARE – By Graham J. McLusky

Introduction

Free software is software, which can be used in any way, copied, studied, modified, and redistributed with or without modifications. It is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. Often it is referred to as "Open Source".

Free software is often, but not always, distributed or available free of a purchase price. It is not always without restriction, but restrictions are only permitted if they are trivial or are set at the minimum necessary to ensure that further recipients also receive these freedoms. It is quickly maturing and becoming part of the rich and complex information society and is about generating revenue from doing work the customer wants and will pay for.

Free software is utterly decoupled from the marketing paradigm that has overtaken the software industry today, because a rational consumer, faced with a choice between products that are essentially free, will always pick the best product for their purposes. It is used extensively today, and by all types of consumers, ranging from business and research institutions to governments. It is behind most of the critical parts of the Internet, it was used (unsuccessfully) as part of the defence in the Microsoft anti-trust trial and the spate of recent IPO's has shown that there's money too. Free software is usually regarded as more reliable than commercial software.

Free Software

Software runs our computers, phones, TVs, media players and more, carrying information and our culture.

Software freedom is based on a special copyright license. Software developers do not have the power to eliminate or override restrictions, but what they can and must do is refuse to impose them as conditions of use of the programme. Software manuals must be free, for the same reasons that software must be free, and because the manuals are in effect part of the software. Free software is usually distributed in a form which can be run by a computer, but which is all meaningless 1s and 0s to a human. Software and other creative works enter the public domain only if the author deliberately surrenders the copyright or if the copyright has expired due to the passage of a legally stipulated period of time.

We maintain this free software definition to show clearly what must be true about a particular software programme for it to be considered free software. While free software by any other name would give you the same freedom, it makes a big difference which name we use: different words convey different ideas. In 1998, some of the people in the free software community began using the term "open source software" instead of "free software" to describe what they do. The Free Software movement and the Open Source movement are today separate movements with different views and goals, although one can and does work together on some practical projects. For the Open Source movement, the issue of whether software should be open source is a practical question, not an ethical one. For the Free Software movement, non-free software is a social problem and free software is the solution.

Freedom – the meaning of the word

There are four kinds of "freedom" for users of "free" software:

1. The freedom to run the programme, for any purpose.
2. The freedom to improve the programme, and release your improvements to the public, so that the whole community benefits.
3. The freedom to redistribute copies so you can help your neighbour.
4. The freedom to study how the programme works, and adapt it to your needs.

Free software is mostly a matter of the users' freedom to run, copy, distribute, study, change and improve the software. A programme is "free software" if users have all of these freedoms. You should also have the freedom to make modifications and use them privately in your own work or play, without even mentioning that they exist.

The freedom to run the program means the freedom for any kind of person or organisation to use it on any kind of computer system, for any kind of overall job and purpose, without being required to communicate about it with the developer or any other specific entity. In this freedom, it is the user's purpose that matters, not the developer's purpose; you as a user are free to run a programme for your purposes and if you distribute it to someone else, she is then free to run it for her purposes, but you are not entitled to impose your purposes on her. In order for these freedoms to be real, they must be irrevocable as long as you do nothing wrong; if the developer of the software has the power to revoke the license, without your doing anything to give cause, the software is not free.

However, certain kinds of rules about the manner of distributing free software are acceptable, when they don't conflict with the central freedoms. Rules about how to package a modified version are acceptable, if they don't substantively block your freedom to release modified versions, or your freedom to make and use modified versions privately. When talking about free software, it is best to avoid using terms like "give away or for free", because those terms imply that the issue is about price, not freedom. The term "free software" has an ambiguity problem: an unintended meaning, "Software you can get for zero price," fits the term just as well as the intended meaning, "software which gives the user certain freedoms." "Free software" has two possible meanings, so you may have to follow it up with "free as in speech/freedom". Unless the software comes with freedom to modify and redistribute, it is not "open source software".

The free software movement aims to secure, for all software users, the freedom to help themselves and the freedom to cooperate with others, when they choose, on a commercial or non-commercial basis. In fact, part of the freedom of free software is that anyone has the right to sell an executable programme released under a free software license at any desired price. In practice the requirements are identical, although because the term "open" doesn't call to mind freedom, it misses the point. For two reasons: because I do not believe that adopting the term "open source" will work to truly bring non-proprietary software to dominance in the marketplace, and more importantly, because I believe freedom in software is so important. Instead, I believe to maximize our chances of success in the marketplace, we should increase the level of emphasis placed on freedom and philosophical concepts of non-proprietary software. I support freedom of speech, but that does not mean I will not speak out and denounce people who say despicable things. But I do believe the non-proprietary software is an ethically better form of software and should be encouraged, and that software freedom is important and needs to be preserved.

GPL = General Public License

There are four fallacies relating to free software which seem to be getting some press at the moment: (a) Giving stuff away doesn't work; (b) No one will create free software applications; (c) The GPL is viral. The GPL is no more "viral" than proprietary software. GPL v2 is a copyleft license, which basically means that the rights granted by the license travel downstream with the code. The most commonly used free software licence is the GNU General Public License (GPL), published by FSF. There is plenty of good software-in-source that is not covered by the GPL. Most Free Software is released under the GPL (General Public License), which means you can modify and redistribute the software, but under the restriction of maintaining credit to previous developers. You also must release the modified software under GPL as well, which means you cannot transform it into Proprietary software. The single biggest contribution of GPL has been the protection against appropriation of Free Software by commercial and proprietary interests. Perl, TCL, Python, all public-domain languages, constitute the "glue" that holds together thousands of Websites * StarOffice, a complete office applications suite programmed in Java, has been released under the GNU GPL by Sun Microsystems. The GPL only obliges you if you distribute software made from GPL'd code, and only needs to be accepted when redistribution occurs. And because no one can ever redistribute without a license, we can safely presume that anyone

redistributing GPL'd software intended to accept the GPL. It can get more complicated than this, of course, especially since the copyright holder of GPL v2-licensed code, for example, can sell it and even use it in closed-source solutions. While the Free Software Foundation's GNU General Public Licenses (GPL) are used in nearly three-quarters of all free and open source software, GPL-based licenses are not the only game in town. In practical terms, copyleft in GPL means that if a GPL-licensed application is licensed as free software, all modified and extended versions of the programme must remain free as well.

Copyright issues

If the programme's license says that you cannot merge in an existing module, such as if it requires you to be the copyright holder of any code you add, then the license is too restrictive to qualify as free. Most free software licenses are based on copyright, and there are limits on what kinds of requirements can be imposed through copyright. If a copyright-based license respects freedom in the ways described above, it is unlikely to have some other sort of problem that we never anticipated (though this does happen occasionally). If a contract-based license restricts the user in an unusual way that copyright-based licenses cannot, and which isn't mentioned here as legitimate, we will have to think about it, and we will probably conclude it is non-free.

Those of us involved in copyright reform find this confusing as all software that is not in the public domain has "proprietors". In the case of FLOSS the copyright holders (proprietors) have licensed their software using terms which protect the rights of those that will then be able to run, copy, distribute, study, change and improve the software. Often there are more copyright holders with peer produced software than with other methods, with the term "proprietary software" only confusing people. So in the proprietary case, not only can work created by the developer be used by someone else, it can actually be taken from the developer, and, on top of that, the copyright holder can sue them for breach. The key assumption underlying copyright policy is that development must necessarily be "big bang" and be done in isolation by "entrepreneurs".

When computer users do not have the freedoms of free software, no-free software, or "proprietary software", uses technical measures to prevent computer users from helping themselves (only distributing binary files, which humans can't read), and uses legal measures to prohibit computer users from helping each other. The copyright holder is legally empowered to exclude all others from copying, distributing, and making derivative works. But most proprietary software companies want more power than copyright alone gives them. These companies say their software is "licensed" to consumers, but the license contains obligations that copyright law knows nothing about. All of those activities are either forbidden or controlled by proprietary software firms, so they require you to accept a license, including contractual provisions outside the reach of copyright, before you can use their works.

Development of free software

Development organisations should encourage and assist project partners in the deployment of software systems that will enable them to "take control of their own destiny" and to reduce their dependence on the developed world. Development organisations should ultimately try to free themselves from the shackles of proprietary software. Development of the GNU tools could thus proceed directly in the environment of university and other advanced computing centres around the world. A free programme must be available for commercial use, commercial development, and commercial distribution. Commercial development of free software is no longer unusual; such free commercial software is very important. They ask us to regard this as legitimate, as part of our community, because some of the money is donated to free software development. That movement does not say users should have freedom, only that allowing more people to look at the source code and help improve it makes for faster and better development. Despite being cheap to obtain the development, testing, and maintenance of the free software you use is not negligible in cost, far from it. Here's what's not free about it: development, testing, maintenance and distribution. Some Free Software users do not seem to realise that Free Software developers have no more obligation to their users to develop or support software than proprietary software developers have obligation for unpaid support or development. These motivations are very important in trying to convince creators to add their works (software, etc) to the commons, and to

help government policy makers to understand these methodologies adequately in their policy development. There are ongoing efforts to help the so-called "World Intellectual Property Organisation" to fulfil its actual mandate (hint: its mandate is not to maximize patents and copyright for the benefit of old-economy monopoly rent seekers) and have meetings to discuss Collaborative Development Models. The market for software is greatly expanded, and the ability for small companies or individual entrepreneurs (the real growth area in our economy) is enhanced because the entry cost for a development environment and other overhead is minimal. "changes the model of software development and distribution to one much like the model our Legal system and its industry uses.

Free Software Foundation

The Free Software Definition of the Free Software Foundation with its four freedoms is the clearest definition existing today. The Free Software Foundation wrote an article titled "Why Free Software" is better than "Open Source", to explain some of the differences in terminology from their perspective.

The Free Software Foundation was founded in October 1985 to handle distribution of GNU software, and to provide a legal infrastructure for GNU and the emerging free software community. You will find a wealth of information about the free software philosophy, the history of the project, and the stands taken by the Free Software Foundation (the organisation behind GNU). As an alternative to the new proprietary programmes, the Free Software Foundation was created as a home for a project; a software licence called the GNU General Public Licence. Although the Free Software Foundation prefers to emphasise philosophical freedoms, its cost-free nature remains one of the main attractions for cash-strapped governments and educational institutions. The philosophy behind the Free Software Foundation and Free Software Movement is that software should be your property and you have the rights to customise it and modify it as per your requirements. The Free Software Foundation (FSF) is a non-profit organisation, which serves as the worldwide headquarters of the Free Software movement. If you find value in Free Software, please consider becoming a member of the Free Software Foundation.

Possible

Although few people can name even one piece of software which they use that has no bugs, defect-free software is possible to create. Although I thought I understood the importance of quality, and took pride in the quality of the software we produced, I never believed that delivering defect-free software was possible. Copyleft advocates also point out that although code released under the GPL cannot be incorporated directly into proprietary software that does not make its full source code freely available, it is still possible to use it effectively with proprietary code. One monopoly dominates most desktop computers, and when competition appears, they threaten patent litigation law suits, or they invent a new secret file format which makes it impossible for users of alternative software to collaborate.

Since the "Open Source" trade marking initiative failed, there is no way to prevent abuse of the term that becomes possible because of the aforementioned misunderstanding. It is not practically possible to use free software in all cases today --- but it should be used when possible. On the other hand, I can think of many small bands of people who, armed with a position they claimed to be morally just, triumphed in the face of almost impossible odds. When writing gateway software of any kind, take pains to disturb the data stream as little as possible --and never throw away information unless the recipient forces you to.

Conclusion

As one person put it, "Open source is a development methodology; free software is a social movement". Commercial development of free software is no longer unusual; such programmes are free commercial software. The hidden cost in free software is the training/retraining of the people that use it, the lack of documentation, and the lack of actual support for the end user, and often a lack of sustainable migration paths to new hardware. For example, free software is generally more secure than proprietary software. Attracting users to free software is not the whole job, just the first step. The amount of developers are growing at an astounding rate, and

the funding for free software is expanding quite drastically, with increasing backing from companies such as Intel, AMD, Nokia, Sun, and many more. The concept of Free Software is often a bit confusing to English speakers as there are multiple meanings of the word free, and the concept is quite different than how the incumbent computer industry works. To repeat the cliché, the free in free software is the "free as in speech, not free as in beer".

The fundamental assumption of free software is that innovation and development comes from minor incremental improvement and massively scaled collaboration and that distribution comes for free (for those of you who've been asleep for the last decade, welcome to the information superhighway!).