

Open Sources: Voices from the Open Source Revolution

Introduction

Chris DiBona, Sam Ockman, and Mark Stone

. . . .

What Is Free Software and How Does It Relate to Open Source?

In 1984, Richard Stallman, a researcher at the MIT AI Lab, started the GNU project. The GNU project's goal was, simply put, to make it so that no one would ever have to pay for software. Stallman launched the GNU project because essentially he feels that the knowledge that constitutes a running program--what the computer industry calls the source code--should be free. If it were not, Stallman reasons, a very few, very powerful people would dominate computing.

Where proprietary commercial software vendors saw an industry guarding trade secrets that must be tightly protected, Stallman saw scientific knowledge that must be shared and distributed. The basic tenet of the GNU project and the Free Software Foundation (the umbrella organization for the GNU project) is that source code is fundamental to the furthering of computer science and freely available source code is truly necessary for innovation to continue.

Stallman worried how the world would react to free software. Scientific knowledge is often in the public domain; it is one function of academic publishing to put it there. With software, however, it was clear that just letting the source code go into the public domain would tempt businesses to co-opt the code for their own profitability. Stallman's answer to this threat was the GNU General Public License, known as the GPL (see Appendix B).

The GPL basically says that you may copy and distribute the software licensed under the GPL at will, provided you do not inhibit others from doing the same, either by charging them for the software itself or by restricting them through further licensing. The GPL also requires works derived from work licensed under the GPL to be licensed under the GPL as well.

When Stallman and others in this book talk about free software, they are really talking about free speech. English handles the distinction here poorly, but it is the distinction between *gratis* and *liberty*, as in "Free as in speech, not as in beer." This radical message (the freedom part, not the beer part) led many software companies to reject free software outright. After all, they are in the business of making money, not adding to our body of knowledge. For Stallman, this rift between the computer industry and computer science was acceptable, maybe even desirable.

What Is Open Source Software?

In the spring of 1997, a group of leaders in the free software community assembled in California. This group included Eric Raymond, Tim O'Reilly, and VA Research president Larry Augustin, among others. Their concern was to find a way to promote the ideas surrounding free software to people who had formerly shunned the concept. They were concerned that the Free Software

Foundation's anti-business message was keeping the world at large from really appreciating the power of free software.

At Eric Raymond's insistence, the group agreed that what they lacked in large part was a marketing campaign, a campaign devised to win mind share, and not just market share. Out of this discussion came a new term to describe the software they were promoting: Open Source. A series of guidelines were crafted to describe software that qualified as Open Source.

Bruce Perens had laid much of the groundwork for the Open Source Definition. One of the GNU project's state goals was to create a freely available operating system that could serve as the platform for running GNU software. In a classic case of software bootstrapping, Linux had become that platform, and Linux had been created with the help of GNU tools. Perens had headed the Debian project, which managed a distribution of Linux that included within the distribution only software that adhered to the spirit of GNU. Perens had laid this out explicitly in a document called the "Debian Social Contract." The Open Source definition is a direct descendant of the "Debian Social Contract," and thus Open Source is very much in the spirit of GNU.

The Open Source Definition allows greater liberties with licensing than the GPL does. In particular, the Open Source Definition allows greater promiscuity when mixing proprietary and open-source software.

Consequently, an Open Source license could conceivably allow the use and redistribution of open-source software without compensation or even credit. As an example you can take great swaths of the Netscape browser source code and distribute it with another, possibly proprietary, program without even notifying Netscape. Why would Netscape wish this? For a number of reasons, but the most compelling is that it gets greater market share for their client code, which works very well with their commercial offerings. In this way, giving away source code is a very good way to build a platform. This is also one of the reasons why the people at Netscape did not use the GPL.

This is not a small issue in the community. Late in 1998, there was an important dispute that threatened to fracture the Linux community. This fracture was caused by the advent of two software systems, GNOME and KDE, each of which aims to build an object-oriented desktop interface. On the one hand, KDE utilized Troll Technology's Qt library, a piece of code that was proprietary, but quite stable and mature. On the other hand, the GNOME people decided to use the GTK+ library, which was a completely free library, though not as mature as Qt.

In the past, Troll Technology would have had to choose between using the GPL and maintaining their proprietary stance. The rift between GNOME and KDE would have continued. With the advent of Open Source, however, Troll was able to change their license to one that met the Open Source definition, while still giving Troll the control over the technology they wanted. The rift between two important parts of the Linux community appears to be closing.

....