Free Software/Open Source

Alley Stoughton
Kansas State University
Spring 2008

Proprietary Software

In the traditional approach to commercial software development and distribution, software is:

Proprietary Software

In the traditional approach to commercial software development and distribution, software is:

- written by centrally controlled, closed groups;

Proprietary Software

In the traditional approach to commercial software development and distribution, software is:

- written by centrally controlled, closed groups;
- released in binary form;

Proprietary Software

In the traditional approach to commercial software development and distribution, software is:

- written by centrally controlled, closed groups;
- released in binary form;
- not accompanied by warranties;

Proprietary Software

In the traditional approach to commercial software development and distribution, software is:

- written by centrally controlled, closed groups;
- released in binary form;
- not accompanied by warranties;
- released under restrictive licenses.

Proprietary Software (Cont.)

For example, here are excepts from the license of the Adobe Reader (`http://www.adobe.com/products/acrobat/acrreula.html`):

> 2.5.1 You may not modify, adapt, translate or create derivative works based upon the Software. You may not reverse engineer, decompile, disassemble or otherwise attempt to discover the source code of the Software except to the extent you may be expressly permitted to decompile under applicable law, it is essential to do so in order to achieve operability of the Software with another software program, and you have first requested Adobe to provide the information necessary to achieve such operability and Adobe has not made such information available.

Proprietary Software (Cont.)

When a user of a proprietary program encounters a bug, he or she may report it to the software developer, and hope that the developer will fix it in the next release of the product or in a patch to the product. Similarly, users who wish for increased or different functionality may suggest product improvements to the developer.

Proprietary Software (Cont.)

When a user of a proprietary program encounters a bug, he or she may report it to the software developer, and hope that the developer will fix it in the next release of the product or in a patch to the product. Similarly, users who wish for increased or different functionality may suggest product improvements to the developer.

But the users are totally dependent on the developer to make needed changes. If the developer declines to make the product support new hardware, for instance, the product may become useless to some users, but there will be nothing that the users can do about this.

Proprietary Software (Cont.)

When a user of a proprietary program encounters a bug, he or she may report it to the software developer, and hope that the developer will fix it in the next release of the product or in a patch to the product. Similarly, users who wish for increased or different functionality may suggest product improvements to the developer.

But the users are totally dependent on the developer to make needed changes. If the developer declines to make the product support new hardware, for instance, the product may become useless to some users, but there will be nothing that the users can do about this.

For example, the Adobe Reader doesn't track changes in PDF files. Because of Adobe's license and practices, there is nothing that can be done about this. This is one of the main reasons that I switched to using Skim (`http://skim-app.sourceforge.net/`), a PDF reader and note-taker for Mac OS X, which is free software.

Free Software

An alternative approach to software development and distribution was pioneered by the Free Software Foundation (`http://www.fsf.org`, `http://www.gnu.org`) and its founder, Richard Stallman.

According to the FSF:

> “Free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer.”

Free Software (Cont.)

According to the FSF:

> Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:

Free Software (Cont.)

According to the FSF:

> Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
>
> - The freedom to run the program, for any purpose.

Free Software (Cont.)

According to the FSF:

> Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
>
> - The freedom to run the program, for any purpose.
> - The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this.

Free Software (Cont.)

According to the FSF:

> Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
>
> - The freedom to run the program, for any purpose.
> - The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this.
> - The freedom to redistribute copies so you can help your neighbor.

Free Software (Cont.)

According to the FSF:

> Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software. More precisely, it refers to four kinds of freedom, for the users of the software:
>
> - The freedom to run the program, for any purpose.
> - The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this.
> - The freedom to redistribute copies so you can help your neighbor.
> - The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. Access to the source code is a precondition for this.

Copyleft

Should someone be able to incorporate free software into a proprietary product?

Copyleft

Should someone be able to incorporate free software into a proprietary product?

In some sense, this is appropriate, since the software was released without restrictions, and since it will lead to more users benefiting from using the software.

Copyleft

Should someone be able to incorporate free software into a proprietary product?

In some sense, this is appropriate, since the software was released without restrictions, and since it will lead to more users benefiting from using the software.

But another point of view is that this would violate the spirit of openness and cooperation in which the original software was created and distributed.

Copyleft

Should someone be able to incorporate free software into a proprietary product?

In some sense, this is appropriate, since the software was released without restrictions, and since it will lead to more users benefiting from using the software.

But another point of view is that this would violate the spirit of openness and cooperation in which the original software was created and distributed.

The FSF proposed the notion of “copyleft” as a way to stop free software from being used in proprietary products.

> Copyleft is a general method for making a program free software and requiring all modified and extended versions of the program to be free software as well.

Software licensed under the FSF’s GNU General Public License (GPL) is copylefted.

More on Copylefting

Copylefting software leads to more free software, since if an individual, group or company produces a program that makes use of free software, they must either:

More on Copylefting

Copylefting software leads to more free software, since if an individual, group or company produces a program that makes use of free software, they must either:

- distribute their program under the same terms as the original software; or

More on Copylefting

Copylefting software leads to more free software, since if an individual, group or company produces a program that makes use of free software, they must either:

- distribute their program under the same terms as the original software; or
- not distribute it at all.

More on Copylefting

Copylefting software leads to more free software, since if an individual, group or company produces a program that makes use of free software, they must either:

- distribute their program under the same terms as the original software; or
- not distribute it at all.

For example, this lead to a free C++ compiler (developed as a front-end to gcc by an industry consortium that normally makes its work proprietary; now called GNU C++). (See `http://www.gnu.org/philosophy/pragmatic.html` for details.)

Making Money out of Free Software

Individuals and companies can make money out of free software by:

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;
- providing education and other support for users;

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;
- providing education and other support for users;
- writing books about free software;

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;
- providing education and other support for users;
- writing books about free software;
- developing custom free software for clients, who may later release them;

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;
- providing education and other support for users;
- writing books about free software;
- developing custom free software for clients, who may later release them;
- selling advertising on free software-related WWW sites.

Making Money out of Free Software

Individuals and companies can make money out of free software by:

- selling distributions of free software;
- providing education and other support for users;
- writing books about free software;
- developing custom free software for clients, who may later release them;
- selling advertising on free software-related WWW sites.

E.g., Red Hat Inc. (`http://www.redhat.com`) has built a large and profitable business out of distributing and supporting Linux.

Free Software Development

Free Software leads to a less centralized way of producing software.

Free Software Development

Free Software leads to a less centralized way of producing software.

Sometimes this results in multiple versions of programs (e.g., different versions of Emacs). This seems to rarely be confusing, in practice.

Free Software Development

Free Software leads to a less centralized way of producing software.

Sometimes this results in multiple versions of programs (e.g., different versions of Emacs). This seems to rarely be confusing, in practice.

But much free software is produced using an approach in which there are principal developers and a group of users/co-developers who find bugs, make suggestions, provide code improvements. Sometimes this leads to faster development of high quality software. It's crucial that the principal developers have high standards.

The Development of

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones. This has been brewing
since april, and is starting to get ready. I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things).
I've currently ported bash(1.08) and gcc(1.40), and things seem to work.
This implies that I'll get something practical within a few months, and
I'd like to know what features most people would want. Any suggestions
are welcome, but I won't promise I'll implement them :-)
Linus (torvalds@kruuna.helsinki.fi)
```

The Development of Linux

```
From: torvalds@klaava.Helsinki.FI (Linus Benedict Torvalds)
Newsgroups: comp.os.minix
Subject: What would you like to see most in minix?
Summary: small poll for my new operating system
Message-ID: <1991Aug25.205708.9541@klaava.Helsinki.FI>
Date: 25 Aug 91 20:57:08 GMT
Organization: University of Helsinki

Hello everybody out there using minix -
I'm doing a (free) operating system (just a hobby, won't be big and
professional like gnu) for 386(486) AT clones. This has been brewing
since april, and is starting to get ready. I'd like any feedback on
things people like/dislike in minix, as my OS resembles it somewhat
(same physical layout of the file-system (due to practical reasons)
among other things).
I've currently ported bash(1.08) and gcc(1.40), and things seem to work.
This implies that I'll get something practical within a few months, and
I'd like to know what features most people would want. Any suggestions
are welcome, but I won't promise I'll implement them :-)
Linus (torvalds@kruuna.helsinki.fi)
```

The Development of Linux (Cont.)

```
PS. Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT protable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's all I have :-(.
```

The Development of Linux (Cont.)

```
PS. Yes - it's free of any minix code, and it has a multi-threaded fs.
It is NOT protable (uses 386 task switching etc), and it probably never
will support anything other than AT-harddisks, as that's all I have :-(.
```

Seventeen years later, (GNU/)Linux has become a mature operating system:

- Linux market share in new server sales is about 20%, as compared to 70% for Windows and 10% for Unix, although the Linux community disputes these numbers, saying that many Linux servers weren't shipped as such.
- Linux desktop market share is about 1%, with Mac OS X at about 7%, and Windows at 92%, although the same caveat applies here. One hopeful sign for Linux is that Dell is now offering the Ubuntu Linux distribution on desktops.

The Cathedral and the Bazaar

In a draft of his book *The Cathedral and the Bazaar* (O'Reilly & Associates, 2001, ISBN 0596001088), Eric S. Raymond wrote about the success of Linux and other free software projects:

> Linux overturned much of what I thought I knew. I had been preaching the Unix gospel of small tools, rapid prototyping and evolutionary programming for years. But I also believed there was a certain critical complexity above which a more centralized, a priori approach was required. I believed that the most important software (operating systems and really large tools like the Emacs programming editor) needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time.

The Cathedral and the Bazaar

Raymond continued:

> Linus Torvalds's style of development - release early and often, delegate everything you can, be open to the point of promiscuity - came as a surprise. No quiet, reverent cathedral-building here – rather, the Linux community seemed to resemble a great babbling bazaar of differing agendas and approaches (aptly symbolized by the Linux archive sites, who'd take submissions from anyone) out of which a coherent and stable system could seemingly emerge only by a succession of miracles.

Raymond notes that Torvalds's people and organizational skills were key to the success of Linux.

Raymond wrote about the circumstances in which this style of software development can succeed, focusing on a case study (Fetchmail (`http://fetchmail.berlios.de/`)) in which he was the lead developer.

Open Source

Here is how the term "open source" was coined, according to the WWW page of the Open Source Initiative (`http://www.opensource.org`):

- The "open source" label itself came out of a strategy session held on February 3rd 1998 in Palo Alto, California. The people present included Todd Anderson, Chris Peterson (of the Foresight Institute), John "maddog" Hall and Larry Augustin (both of Linux International), Sam Ockman (of the Silicon Valley Linux User's Group), and Eric Raymond.

Open Source

Here is how the term “open source” was coined, according to the WWW page of the Open Source Initiative (`http://www.opensource.org`):

- The “open source” label itself came out of a strategy session held on February 3rd 1998 in Palo Alto, California. The people present included Todd Anderson, Chris Peterson (of the Foresight Institute), John “maddog” Hall and Larry Augustin (both of Linux International), Sam Ockman (of the Silicon Valley Linux User’s Group), and Eric Raymond.
- We were reacting to Netscape’s announcement that it planned to give away the source of its browser. One of us (Raymond) had been invited out by Netscape to help them plan the release and followon actions. We realized that the Netscape announcement had created a precious window of time within which we might finally be able to get the corporate world to listen to what we have to teach about the superiority of an open development process.

Open Source (Cont.)

- We realized it was time to dump the confrontational attitude that has been associated with “free software” in the past and sell the idea strictly on the same pragmatic, business-case grounds that motivated Netscape. We brainstormed about tactics and a new label. “Open source,” contributed by Chris Peterson, was the best thing we came up with.

Open Source (Cont.)

- We realized it was time to dump the confrontational attitude that has been associated with “free software” in the past and sell the idea strictly on the same pragmatic, business-case grounds that motivated Netscape. We brainstormed about tactics and a new label. “Open source,” contributed by Chris Peterson, was the best thing we came up with.
- Over the next week we worked on spreading the word. Linus Torvalds gave us an all-important imprimatur :-) the following day. Bruce Perens got involved early, offering to trademark “open source” and host this web site. Phil Hughes offered us a pulpit in Linux Journal. Richard Stallman flirted with adopting the term, then changed his mind.

Open Source (Cont.)

Here is some of what the Open Source Initiative's says about the benefits of Open Source:

- The basic idea behind open source is very simple. When programmers on the Internet can read, redistribute, and modify the source for a piece of software, it evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.

Open Source (Cont.)

Here is some of what the Open Source Initiative's says about the benefits of Open Source:

- The basic idea behind open source is very simple. When programmers on the Internet can read, redistribute, and modify the source for a piece of software, it evolves. People improve it, people adapt it, people fix bugs. And this can happen at a speed that, if one is used to the slow pace of conventional software development, seems astonishing.
- We in the open-source community have learned that this rapid evolutionary process produces better software than the traditional closed model, in which only a very few programmers can see source and everybody else must blindly use an opaque block of bits.

Open Source (Cont.)

- The open-source model has a lot to offer the business world. It's a way to build open standards as actual software, rather than paper documents. It's a way that many companies and individuals can collaborate on a product that none of them could achieve alone. It's the rapid bug-fixes and the changes that the user asks for, done to the user's own schedule.

Open Source (Cont.)

- The open-source model has a lot to offer the business world. It's a way to build open standards as actual software, rather than paper documents. It's a way that many companies and individuals can collaborate on a product that none of them could achieve alone. It's the rapid bug-fixes and the changes that the user asks for, done to the user's own schedule.
- The open-source model also means increased security; because code is in the public view it will be exposed to extreme scrutiny, with problems being found and fixed instead of being kept secret until the wrong person discovers them. And last but not least, it's a way that the little guys can get together and have a good chance at beating a monopoly.

The FSF's Views on Free Software/Open Source

The FSF offers its views on the relationship between free software and open source (`http://www.gnu.org/philosophy/free-software-for-freedom.html`):

- The fundamental difference between the two movements is in their values, their ways of looking at the world. For the Open Source movement, the issue of whether software should be open source is a practical question, not an ethical one. As one person put it, "Open source is a development methodology; free software is a social movement." For the Open Source movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution.

The FSF's Views on Free Software/Open Source

The FSF offers its views on the relationship between free software and open source (`http://www.gnu.org/philosophy/free-software-for-freedom.html`):

- The fundamental difference between the two movements is in their values, their ways of looking at the world. For the Open Source movement, the issue of whether software should be open source is a practical question, not an ethical one. As one person put it, "Open source is a development methodology; free software is a social movement." For the Open Source movement, non-free software is a suboptimal solution. For the Free Software movement, non-free software is a social problem and free software is the solution.

- We disagree on the basic principles, but agree more or less on the practical recommendations. So we can and do work together on many specific projects. We don't think of the Open Source movement as an enemy. The enemy is proprietary software.

SourceForge

A good place to look for information about free software/open source is SourceForge (`http://web.sourceforge.com/`):

- SourceForge's media and e-commerce web sites connect millions of influential technology professionals and enthusiasts each day. Combining user-developed content, online marketplaces and e-commerce, SourceForge is the global technology community's nexus for information exchange, goods for geeks, and open source software distribution and services. The network of web sites enables advertisers to efficiently reach a large, highly qualified audience of buyers. SourceForge's network serves more than 33 million unique visitors each month from around the world.

SourceForge (Cont.)

- It includes top web sites, like SourceForge.net, the world's largest open source software development and distribution environment; Slashdot, the web destination that pioneered community generated content; and ThinkGeek, the online bazaar that features cool stuff for techno-enthusiasts. Other sites in the network, include: Linux.com, freshmeat.net, ITManagersJournal and NewsForge. SourceForge's unique combination of user-developed content, clever e-commerce and online marketplaces make it the most trusted, credible venue for dialogue and exchange with the global technology community.

Freshmeat

According to `http://freshmeat.net`:

> freshmeat maintains the Web's largest index of Unix and cross-platform software, themes and related "eye-candy", and Palm OS software. Thousands of applications, which are preferably released under an open source license, are meticulously cataloged in the freshmeat database, and links to new applications are added daily. Each entry provides a description of the software, links to download it and to obtain more information, and a history of the project's releases, so readers can keep up-to-date on the latest developments.

`Freshmeat.net` lists over 40,000 projects. 63.19% of the software listed at freshmeat is released under the GPL (6.6% is under a less restictive version of GPL, the LGPL).

SourceForge

According to `http://sourceforge.net`:

> `SourceForge.net` is the world's largest Open Source software development web site, hosting more than 100,000 projects and over 1,000,000 registered users with a centralized resource for managing projects, issues, communications, and code. SourceForge.net has the largest repository of Open Source code and applications available on the Internet, and hosts more Open Source development products than any other site or network worldwide. SourceForge.net provides a wide variety of services to projects we host, and to the Open Source community.

`SourceForge.net` hosts over 170,000 projects.

Key Free Software/Open Source Successes

In addition to Linux, here are two free software/open source success stories:

- As of August, 2007, 48.4% of all web sites used the Apache web server open source software; Apache's closest competitor was Microsoft-IIS at 36.2%. Trends show Microsoft closing the gap, and perhaps surpassing Apache soon, however.

Key Free Software/Open Source Successes

In addition to Linux, here are two free software/open source success stories:

- As of August, 2007, 48.4% of all web sites used the Apache web server open source software; Apache's closest competitor was Microsoft-IIS at 36.2%. Trends show Microsoft closing the gap, and perhaps surpassing Apache soon, however.
- As of January, 2008, the Mozilla Foundation's Firefox browser (which is descended from the Netscape browser) had a 15% share of browser usage, with Microsoft's Internet Explorer at 75% and Mac OS X's Safari at 6%. Firefox's usage share looks likely to continue growing.

Threats to Free Software/Open Source

The threats to free software/open source seem to be mainly non-technical:

- Hardware manufacturers (like Intel) are often slow to make specifications for chips available to open source developers, while promptly making those specifications available to Microsoft. But in 2004, Intel announced that, in the future, proprietary Linux drivers (but not complete specifications) for their hardware will arrive during the same release cycle as Windows drivers, though not necessarily the same day.

Threats to Free Software/Open Source

The threats to free software/open source seem to be mainly non-technical:

- Hardware manufacturers (like Intel) are often slow to make specifications for chips available to open source developers, while promptly making those specifications available to Microsoft. But in 2004, Intel announced that, in the future, proprietary Linux drivers (but not complete specifications) for their hardware will arrive during the same release cycle as Windows drivers, though not necessarily the same day.
- Software patents and copyrights are being used by companies to threaten Linux and other open source efforts. According to Linus Torvalds, "The things that tend to worry me are software patents. When nontechnical issues can be used to stop software development—that for me is the scariest part."

Threats to Free Software/Open Source

- A February 22, 2008, *New York Times* article writes:

  Seeking to satisfy European antitrust officials, Microsoft said on Thursday that it would open up and share many more of its technical secrets with the rest of the software industry and competitors.

  Microsoft executives, in a conference call, characterized the announcement as a “strategic shift” in the company’s business practices and its handling of technical information. They also portrayed the moves as only partly a nod to the continuing challenge Microsoft faces from Europe’s antitrust regulators.

Free Software/Open Source Resources

On the WWW page

`http://people.cis.ksu.edu/~stough/free/`

I've collected together links to various resources on free software/open source. You can also find the slides for this talk there.

Possible Discussion Questions

- Should critical software applications, like heart pacemaker software, be developed using the open source model?
- What is stopping Linux from capturing a significant share of the desktop market? Could this change?
- Should CIS encourage student involvement in open source projects?