Open Source Software: The Engine Behind an Internet Start-up Company

Dan R. Lipşa and Robert S. Laramee

Visual and Interactive Computing Group,
Department of Computer Science,
Swansea University, Swansea, UK
{d.lipsa, r.s.laramee}@swansea.ac.uk

May 27, 2010

Abstract

Open Source Software (OSS) is a ubiquitous term in this day and age. We identify and describe what Open Source Software is, and explain its relationship with Free Software. We then present a case study: two ways of using Open Source Software in a business environment: internal use and use in a proprietary product as a standalone module or library. Both of these were successfully pursued in a real company. Our case study includes the use of Open Source Software in an Internet start-up's IT infrastructure (internal use), as well as in their product. We demonstrate how Open Source Software can provide real value to businesses that are willing to exploit its advantages.

Keywords: open source software (OSS), free software, OSS IT infrastructure use, OSS product use.

1 Free Software and Open Source Software

No description of Open Source Software can start without mentioning Richard Stallman, Free Software Foundation (FSF) and the GNU Project. Richard Stallman is the founder of Free Software Foundation – a tax-exempt charity that raises funds for work on the GNU Project. The GNU Project project implements an operating system similar to Unix but this project is Free Software and has no Unix code. GNU is a recursive acronym that stands for "GNU's not Unix".

The GNU project started in 1983, with an email to a Unix newsgroup in which Richard Stallman states that he is going to write a complete Unix-compatible software system and share with everybody. He asks for contributions of time, money, programs and equipment. With the help of thousands of programmers from around the world, and with the arrival of of Linux, Richard Stallman succeeded in doing just that. He is the initial developer for GNU C Compiler, GNU Emacs, GNU debugger, and GNU Make and FSF developed the Bourne Again Shell (bash) and the C library.

1.1 Free Software

Richard Stallman sees software as information, and he believes everyone should have the freedom to use it and to learn from it. In particular, for a given application anyone should be able to run it for any purpose, to study how the program works and adapt it to their needs, redistribute copies in order to help a colleague, improve the program and release the improvements. The the whole community should benefit. A common misconception that FSF tries to correct, is that "free software" means no money for your work. Free refers to freedom, and free is interpreted as in "free speech" not as in "free lunch".

1.2 Copyleft

Copyleft is the use of a license to protect the rights of free software (as defined in Section 1.1) such that continues to be free software. As an example of what can happen if a free software project is not protected, FSF provides the following story of the X Windows System.

The X Windows System, a windowing system for Unix developed at MIT and released as free software with a permissive license, was adopted by many software companies. They shipped their versions of X without sources, so in those releases, the software no longer qualified as free software. The users didn't have the same freedom as they had for the initial release of X. Copyleft was invented to prevent this.

Copyleft uses copyright law, but flips it over to serve the opposite of its original purpose. Instead of keeping the software proprietary, it becomes a means to keep the software free. Copyleft, gives anyone permission to run, copy, modify the program, and distribute modified versions – but not permission to add restrictions of their own. The term copyleft comes from a letter sent to Richard Stallman by Don Hopkins, in which the following phrase appears: "Copyleft – all rights reversed" [16].

1.3 GNU General Public License (GPL) and GNU Lesser (Library) GPL (LGPL)

GNU GPL and LGPL are some of the most popular licenses used for free software. They are both defined by the FSF. Both GNU GPL and LGPL are "free software" licenses as defined in Section 1.1, but differ in their "copyleft" strength. The GPL license is more strict than LGPL. In practical terms this comes down to the fact that only GPL programs are allowed to link with GPL protected libraries, LGPL allows proprietary programs to link with them, as well. It is still possible, to release proprietary software that calls an executable that is protected by GPL [4]. Most of libraries on GNU/Linux systems are protected by LGPL, or less strict licenses, which means that the user may release proprietary programs on GNU/Linux, and link with the libraries available. Many companies have done so [18].

Notably, a Microsoft vice-president comments on GPL that it is against “intellectual property rights” and it contradicts the American Way [9, 10, 17]. FSF responds [9, 17] that copyright cannot justify denying people important freedoms and that Free Software Movement embodies the American Way by promoting freedom, community and voluntary cooperation.

1.4 Open Source Software

Open Source Software is a term promoted by the Open Source Initiative (OSI) [13], a non-profit organization launched in 1998. In their own words, the movement, is a marketing campaign to turn around the negative image free software had outside the hacker community, and argue for free software on pragmatic grounds of reliability, cost and business risks.

Development in the Open Source world happens at an astonishing pace compared with the conventional software development [15]. This is a consequence of the fact that source code is freely available and can be changed. Anyone can find and fix bugs and add their own improvements to the code. This rapid evolutionary process produces better software than the traditional closed model. For this reason and because of the smaller cost it make business sense to choose open source software and contribute to its development.

The official definition of open source software is very close to how FSF defines free software. The two movements differ in the reason they believe why people should adopt open source/free software. For FSF, the belief is that people want and deserve freedom, as defined in Section 1.1. For OSI the belief is that software produced in an open source environment is technically superior.

From now on, we will use only the term Open Source Software, because it appears to be much more popular than Free Software in the written press.

2 Internal Use of Open Source Projects: Building an IT infrastructure

We present a case study of an IT infrastructure for a software company. We describe the functions of each server and then describe the Open Source programs that were used in a real company to carry out those functions.

Figure 1 represents a typical IT infrastructure for a software company, with several of the most common servers. We have a Fire Wall, that separates unwanted traffic from the company’s Intranet. The Fire Wall blocks all the traffic from outside except for email, web, Virtual Private Network (VPN) and Domain Name System (DNS) traffic. We have a web server that responds to users requesting web pages. Web pages, can be generated dynamically, and many times in generating dynamic pages, information is pulled up from a database served

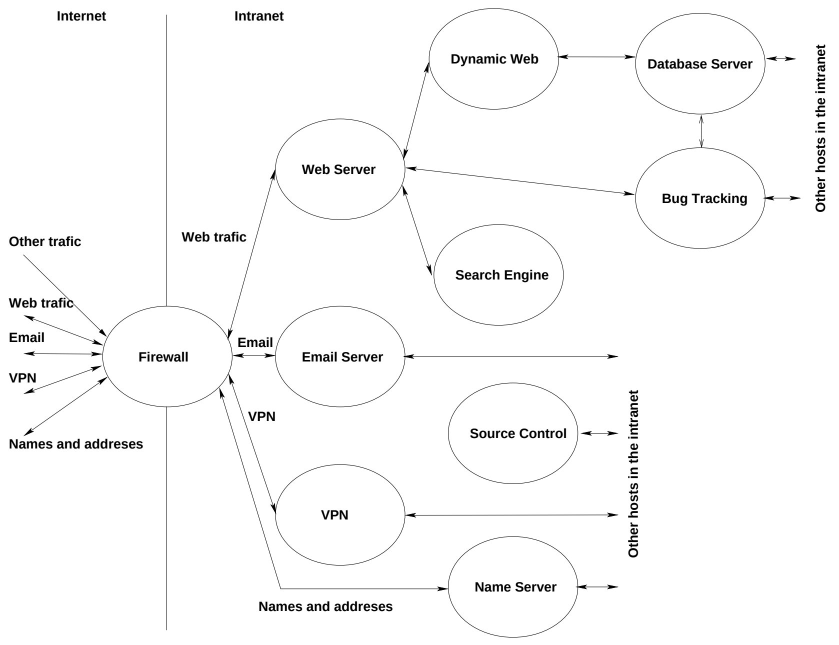

Figure 1: An IT infrastructure of a typical software company uses a Web Server, a Dynamic Web module to creates dynamic web pages from database information, a Database Server, and a Search Engine. It has a Bug Tracking server for reporting and tracking bugs and a Source Control system for keeping track of the software source files. It uses an Email Server, a Virtual Private Network (VPN) server and a Name Server.

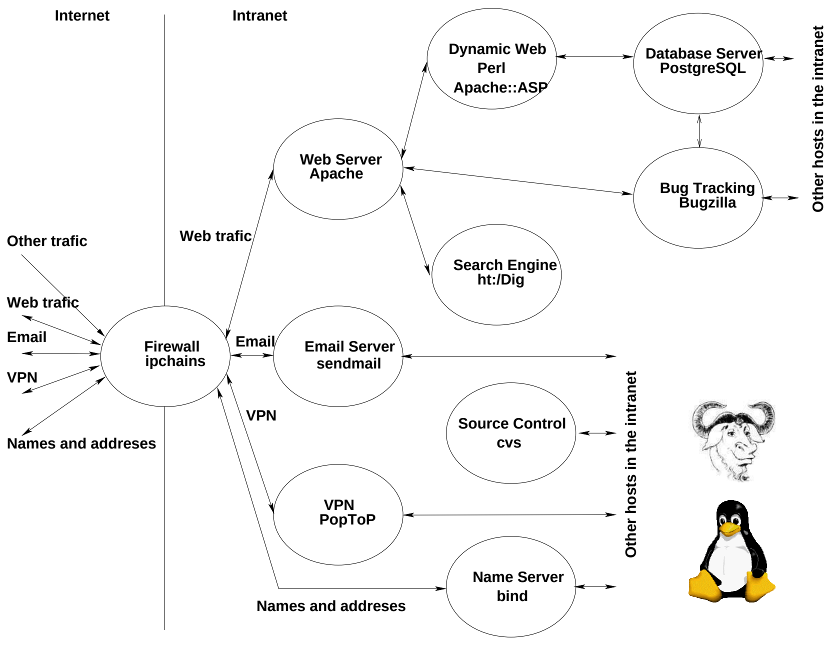

Figure 2: A real case-study IT Infrastructure. All the modules required in a typical software company IT infrastructure are provided by OSS programs. The GNU head is the mascot of the GNU Project. The penguin is the mascot of the Linux operating system.

by a Database Server. Many web sites allow searching through their pages, and for this they use a Search Engine.

An Email server is responsible for moving email from the source to the destination computer over the Internet. A Virtual Private Network (VPN) server, allows people outside of the company's Intranet to securely connect to company's network. Lastly, we have a Name Server (Domain Name System - DNS), that allows name assignment to internal computers.

A software company, would have a couple of more servers.

A Bug Tracking System is a database for bugs. It enables the submission of bug reports for a product, changing the status of a bug report (typical statuses are: resolved, in progress, won't solve and duplicate) and assigning bugs to different developers. It also allows the generation of reports based on different criteria such as: all the bugs for a particular version of the program, all the bugs assigned to a particular developer, all the bugs solved for a particular version of the program.

A Source Control System is used to manage releases and control the concurrent editing of source files among multiple developers. It works by storing each version of a source file and allowing retrieval of any particular version.

Figure 2 represents a case-study IT infrastructure for a real company. Observe, the IT Infrastructure matches the infrastructure in figure 1 just that all the servers are open source software.

The penguin in the picture is the official mascot of Linux, which was selected by Linus Torvalds to represent the image he associates with the operating system he created.

The GNU head is the official mascot for the GNU Project. This is how they describe it on their web page: "A handsome GNU Head with typical beard and smart-looking curled horns. He or she appears to be smiling contently with his work as of yet, but still gazes off into the distance" [5]

We describe the best known open source servers used in our IT infrastructure case study:

Operating System *Linux* is a widely used, modern operating system that was initially created as a hobby by a young student, Linus Torvalds, at the University of Helsinki in Finland. The first version 0.02 was released in 1991, and version 1.0 was release in 1994 [7]. Linux is released under GPL. Netcraft Web Server Survey [11] found that Linux runs on about 29.9% of the active web sites, Microsoft OS runs on about 28.32%, and Solaris is third with 16.33%. Companies like IBM, Oracle and Dell fully support Linux.

Web Server *Apache* is a powerful, full-featured, efficient, and freely-available Web server. Apache is also the most popular Web server on the Internet. A Netcraft Web Server Survey [11] found that over 59% of the web sites on the Internet are using Apache, thus making it more widely used than all other web servers combined. The Apache web server is based on National Center for Supercomputing Applications (NCSA), University of Illinois, Urbana-Champaign, public domain HTTP daemon, and the first release was in 1995. It is released under a simple, non-copyleft open source software license. Examples of sites which run Apache for their web sites are: Apple `www.apple.com`, Financial Times `www.ft.com`, Sony `www.sony.com`, Palm `www.palm.com`, Cnet `www.cnet.com` and Amazon `www.amazon.com`.

Perl Perl started as a Unix administration tool, taking its best features from languages like C, awk, sed and sh. Perl was created by Larry Wall, and version 1.0 was released in 1987 [2]. It is released under GPL. Perl is a popular web programming language, due to its facility with text manipulation and rapid development cycle. Major sites, such as Amazon [1, 12], use Perl for software development. Perl has been called "the duct-tape of the Internet" [12].

3 Use of Open Source Modules in a Proprietary Product

We describe a real application and how it benefits from the use of Open Source modules and libraries. We analyze the Open Source license issues that the

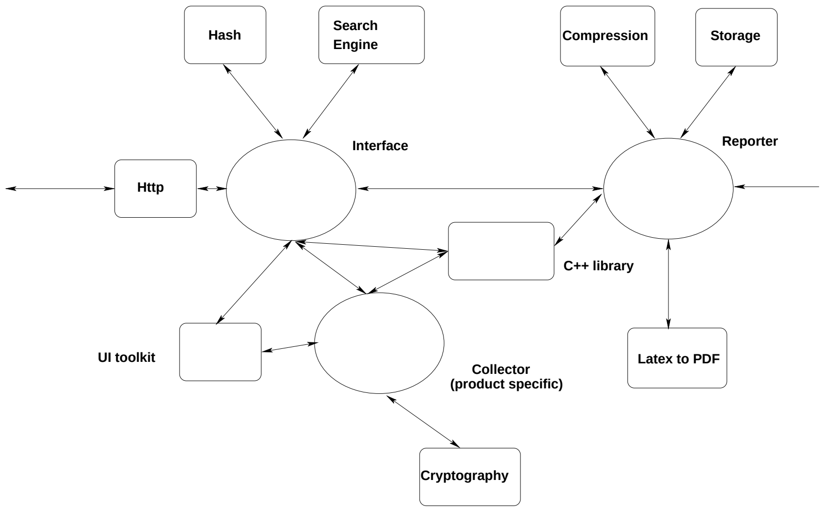

Figure 3: A Case Study Application. It gathers software configuration data and produces customized reports of that data. It tracks software configurations changes and enforces standard configuration compliance. It consists of three modules represented as ovals (Interface, Collector and Reporter). Each module uses one or more libraries represented as rounded rectangles.

user has to pay attention to when using Open Source modules in a proprietary product.

The application depicted in Figure 3 is a software configuration reporting tool. It gathers software configuration data, and produces customized HTML and PDF reports of that data. The application also allows, tracking software configuration changes, by using a "difference" report. This report, takes as input two configuration data snapshots and analyses the difference between them to see what changed. Another function of the application is enforcing standard configuration compliance. This is done by using a configuration data snapshot for a server as reference and testing the difference between that reference and configuration data snapshots for other servers. In this way, we can test if all the servers are configured in the same way.

The interface module interacts with the user and drives all the operations in the application. To display the user interface (UI), it uses an UI toolkit library. The interface connects to the company's website, and sends the system ID (an unique identifier of the machine the application was installed on), the user name and the password (obtained when the application was purchased) and downloads the license required to run the application. Also, by connecting to the company's website, it can check and automatically upgrade the application if a new upgrade exists.

To maintain the integrity of the license, the program calculates a hash value of the license together with a secret key when the license is created, it sends the license together with the hash value through the network, and recalculates the hash value when the license is applied to the program. To accept the license, those two hash values have to be the same.

The user is allowed to search through HTML reports. For this we use a search engine (the same one used in our IT infrastructure).

To collect configuration data, we use a collector module. This module, connects to remote servers and collects configuration data, using product specific means (parsing configuration files, reading registry keys, calling product specific functions). The data is stored as a pair key (name of the configuration data), value. To be able to restrict the volume of data collected as well as to read passwords required to connect to remote machines, the collector utilizes some user input as well. We save the choices the user made in a file, using a cryptography library to encrypt the passwords. We can use that file later when we want to run the program in batch mode, without any user interaction.

To generate the reports in HTML and LaTeX format we use the Reporter module. This reads the data produced by the collector module and stores it in a structure that allows fast searches for configuration keys. The reporter uses a product specific model that describes the correspondence between the data keys and the documentation that has to be generated for every specific key. The reporter uses a compression library to be able to save the storage compressed so that it saves space. Also, it uses a LaTeX to PDF converter so that it generates HTML as well as PDF reports.

In Figure 4, we detail the libraries used in building the application. We use no pattern to flag Open Source modules, the brick wall pattern for proprietary modules and the wave pattern to flag dual license modules. The application uses proprietary, dual license and open source libraries. We describe some of the open source libraries used in the case study application.

Search Engine *ht://Dig* is a Web search and indexing system for a small domain or Intranet [8]. It was developed by San Diego State University, first version being release in 1995. It is release under the GPL.

UI toolkit lib *Qt* provides a platform-independent interface to all central platform functionality: GUI, database access, networking, file handling, etc [3, 14]. The Qt library encapsulates the different functions of different operating systems, providing the application programmer with a single, common interface for all operating systems. The native C functions are encapsulated in a set of well-designed, fully object-oriented C++ classes. The development for Qt started in 1992. Trolltech, the company that sells it, was founded in 1994, and the first commercial version was released in 1995. It is used by KDE desktop environment available on Linux and by the Opera browser.

The Qt license has an interesting history. The library became popular with its use in the KDE desktop and started to be included in Suse, a

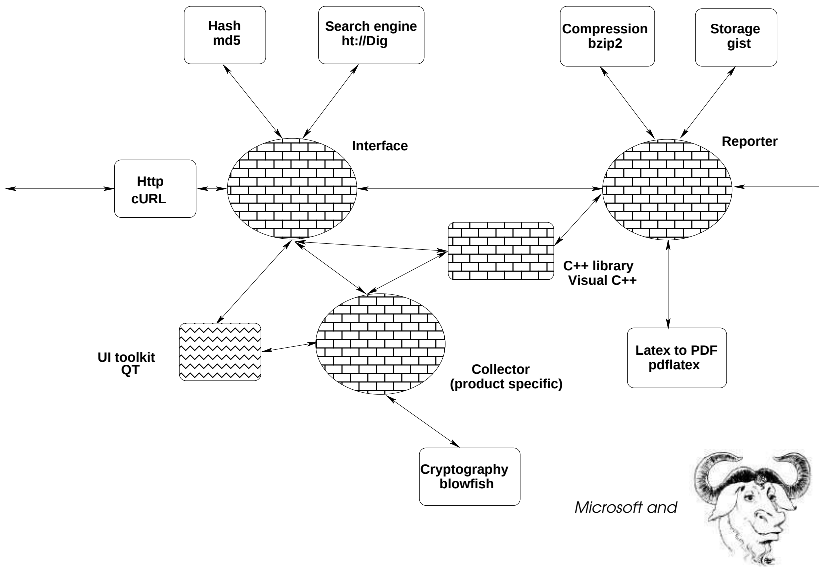

Figure 4: Application with Open Source libraries and modules

German distribution of Linux. Members of the open source community started a campaign against KDE (because of QPL the license used by Qt. The main problem with that license was that Trolltech required people to submit changes to them, and they decide what changes go in) and Red Hat didn't include them in the standard distribution. In 2000, Qt on Unix was release under GPL. One reason is that Qt receives a great deal of exposure because of their use in KDE. They didn't want to throw away this free advertising. Releasing the library under GPL does not prevent it from generating income.

The reason is that GPL doesn't allow proprietary applications to link with a library protected by GPL. So a proprietary program, would have to pay for their use of Qt. Only Open Source programs are allowed to use it free of charge.

Storage lib *Generalized Search Tree C++ Library (GiST)* is a template index structure that makes it easy to implement any type of hierarchical access method [6]. It comes with B-tree and R-tree access methods. It was developed at University of California at Berkeley and it was released as open source software without a copyleft. The first version was released in 1997 and now is included in PostgreSQL, an open source database management system.

4 Conclusions: Why Open Source Software?

We summarize the advantages of using Open Source Software we observed while developing our case study application:

- Open Source Software allows the user to port it and use it on any platform of interest. Proprietary software is usually shipped on specific platforms. If software is required on a platform that is not supported, the user is out of luck.
- The developer can fix problems in Open Source modules. It might not be easy, but it is better than waiting for a fix from a vendor. That fix might not have high priority in a third party's business plan.
- In many cases, an Open Source Software project is the de facto standard for that particular type of application. So, you are using the best application possible. Eg. Apache, Linux, sendmail, bind.
- The up-front cost for Open Source Software is very low. That works very well with small companies with relatively little money.
- Releasing under a open source license can be a very good way of gaining a lot of users (market share) and exploiting free advertising and free testing. The QT case is a very good example of this.

An obvious question in the reader's mind at this point is: OK, but does the adoption of open source development model hurt profits? There are some good examples of where it has not. See Trolltech (the makers of Qt), Red Hat and others. We also note that the ability to generate profit is a challenge to all software companies – not only those that exploit open source software. However, companies are still struggling to accommodate both open source model and the desire to maximize profits. If that is possible for everybody in every situation is still very much an open question.

5 Acknowledgments

This research was partially funded by the Welsh Institute of Visual Computing (WIVC).

References

[1] Amazon.com. Amazon.com Interview: Larry Wall. Online document. Accessed Nov. 28, 2009, `http://www.amazon.com/exec/obidos/tg/feature/-/7137/`.

[2] Elaine Ashton. The Timeline of Perl and its Culture, v3.0_0505. Online document. Accessed Nov. 28, 2009, `http://history.perl.org/PerlTimeline.html`.

[3] Jasmin Blanchette and Mark Summerfield. *C++ GUI Programming with Qt 4*. Prentice Hall, 2008.

[4] Free Software Foundation. Frequently Asked Questions about the GNU Licenses. Online document. Accessed Sep. 26, 2009, `http://www.gnu.org/licenses/gpl-faq.html#MereAggregation`.

[5] Free Software Foundation. The GNU Project and the Free Software Foundation. Online document. Accessed Aug. 31, 2009, `http://www.gnu.org`.

[6] The Generalized Search Tree (GiST). Online document. Accessed Jan. 08, 2010, `http://gist.cs.berkeley.edu`.

[7] Ragib Hasan. History of Linux. Online document. Accessed Aug. 31, 2009, Published Jul. 2002, `https://netfiles.uiuc.edu/rhasan/linux/`.

[8] ht://Dig – Internet Search Engine Software. Online document. Accessed Nov. 28, 2009, `www.htdig.org`.

[9] Bradley M. Kuhn. The GNU GPL and the American Dream. Online Document. Accessed Sep. 26, 2009, `http://www.gnu.org/philosophy/gpl-american-dream.html`.

[10] Ralph Nader and James Love. RE: US v. Microsoft proposed final order. Online document. Accessed Sep. 22, 2009, Published Nov. 5, 2001, `http://www.nader.org/releases/msfinalorder.html`.

[11] Netcraft. Netcraft Web Server Survey. Online document. Accessed August 31 2009, `http://www.netcraft.com/survey`.

[12] Inc. O'Reilly & Associates and Ronin House Ben Smith. The Importance of Perl. Online document. Accessed Nov. 28, 2009, `http://www.oreillynet.com/pub/a/oreilly/perl/news/importance_0498.html`.

[13] Open Source Initiative. Online document. Accessed August 31, 2009, `http://www.opensource.org`.

[14] Qt Development Frameworks a Nokia unit. Online Document. Accessed Aug. 31, 2009, `http://qt.nokia.com`.

[15] Eric S. Raymond. The Cathedral and the Bazaar. Online Document. Accessed Aug. 31, 2009, Published August 02, 2002 `http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/`.

[16] Richard Sallman. The GNU Project. Online document. Accessed Aug. 31, 2009, `http://www.gnu.org/gnu/thegnuproject.html`.

[17] Richard Stallman. The GNU GPL and the American Way. Online document. Accessed Aug. 31, 2009, `http://www.gnu.org/philosophy/gpl-american-way.html`.

[18] Richard Stallman. Why you shouldn't use the Lesser GPL for your next library. Online document. Accessed Aug. 31, 2009, `http://www.gnu.org/licenses/why-not-lgpl.html`.