Software Freedom, Open Software and the Undergraduate Computer Science Curriculum

John E. Howland
Department of Computer Science
Trinity University
715 Stadium Drive
San Antonio, Texas 78212-7200
Voice: (210) 999-7364
Fax: (210) 999-7477
E-mail: jhowland@Trinity.Edu
Web: http://www.cs.trinity.edu/~jhowland/

April 14, 2000

Abstract

The early history of computing records the sharing of computer software in source form. The sharing took place between programmers, and users of programs throughout the computing industry. Later software vendors became rather proprietary with software, using copyright, patent and trade-secret law to restrict the use of programs. In the 1980's, Richard Stallman led a small group of people who were interested in reclaiming the early software freedom enjoyed by programmers and users. These efforts have grown into what is now known as the *Open Source Revolution* and produced such remarkable products as `Emacs`, `gcc` and Linux. The relationship of software freedom, open sources and the undergraduate computer science curriculum are presented. [1]

Subject Areas: Computer Science Education, Computer Science Curriculum.
Keywords: GNU Software, Software Freedom, Open Software, Open Sources.

[1]This refereed paper was published in the the Journal of Computing in Small Colleges, Volume 15, Number 3, March 2000, Pages 293-301. This paper was presented at the CCSC South-Central Conference, April 14, 2000, Texas A&M University Corpus Christi, Corpus Christi Texas.

1 Introduction

Recently, the Linux operating system has become a topic for discussion, even amongst persons outside the computing industry, as the public stock offering of RedHat Software has made three persons associated with RedHat billionares (at least on paper). RedHat Software distributes a version of the Linux system which is the end result of the efforts of thousands of programmers who have given freely of their time and energy to develop software which is distributed under a license which places few restrictions on those who use the software. The software itself is available at no cost on the Internet or, for a nominal fee, on convenient CDROM packages which include documentation and installation manuals and various levels of customer support.

How is it possible that a company could have a stock market value of several billion dollars when the main product sold by the company contains a license which requires that the product must also be given away at no charge? Bob Young, president of Red Hat Software, says that he is selling bits on the basis of brand-name endorsement. Young says that he's in the same kind of business that Evian, a company selling "millions of dollars of French tap water" solely on the strength of its brand and an "irrational fear that the water coming from your tap is not to be trusted". [HAC 1999]

There is something more fundamental supporting the Linux software movement and RedHat Software than selling water which is usually free at the tap. It is the concept of software freedom; not free software, but software whose license does not restrict a user's freedom in any significant way.

2 Software Freedom

The early history of computing records the sharing of computer software in source form. The sharing took place between programmers, and users of programs throughout the computing industry. Later software vendors became rather proprietary with software, using copyright, patent and trade-secret law to restrict the use of programs.

Richard M. Stallman was the first to attempt to deal with restrictive approaches to software licensing. He took a radical personal stand of quitting his job and beginning the job of producing a complete operating system which would be freely available to anyone who wanted to use the system, including source code. In a recent interview, [Lin 1999] Stallman questioned the validity of *intellectual property* in software which seems to value the rights of software producers over software users. Stallman asks: "If you say that the topic you are talking about is property, that presupposes an answer to the most important question: How should these things be treated? Should they be somebody's property or not?"

2.1 Software Liberty

Hackvän [HAC 1999] argues that the word *free* denotes *gratis* as in "free beer" far more readily than it denotes *liberty* as in "free speech". But Stallman is talking about liberty when he describes [DiB 1999] four freedoms for software users:

- Freedom to run a program for any purpose.
- Freedom to modify a program to suit your needs. To make this freedom effective in practice, you must have access to the source code, since making changes in a program without having the source code is exceedingly difficult.
- Freedom to redistribute copies of a program, either gratis or for a fee.
- Freedom to distribute modified versions of the program, so that the community can benefit from your improvements.

2.2 The GNU Software License

Stallman decided that his operating system would be compatible with the widely used proprietary Unix operating system. He used the recursive acronym GNU which stood for "GNU is not Unix" to name the system. When Stallman began writing his operating system he started with an aborted attempt of producing a C compiler which was based on the Lawrence Livermore Laboratory Pastel compiler. This project failed because the internal structure of the compiler required too much stack memory for the Motorola 68000 computer architecture. Parts of this work were later included in the program now known as gcc. Stallman then began work on an extensible editor known as Emacs. When finished, not being employed, he looked for a way to distribute the program without restricting the freedom of those who would use it. A friend gave him an idea in the phrase "Copyleft – all rights reversed". Stallman [DiB 1999] writes about the GNU Software License: "The central idea of copyleft is that we give everyone permission to run the program, copy the program, modify the program, and distribute modified versions–but not the permission to add restrictions of their own. Thus, the crucial freedoms that define *free software* are guaranteed to everyone who has a copy; the become inalienable rights.

For an effective copyleft, modified versions must also be free. This ensures that work based on ours becomes available to our community if it is published"

The complete text of the GNU software license may be found in any GNU software package or at http://www.gnu.org/copyleft/gpl.html

2.3 The GNU System

Stallman originally planned that the GNU system would be developed and released as a complete system. However, as many people began to contribute to

the various parts of the system, these parts were released as preferred replacement parts of, or in addition to, proprietary Unix systems. For example, gcc became a preferred C compiler on most Unix systems and Emacs was a widely used editor as were GNU versions of nearly all the Unix commands and utilities. The only missing major component in the GNU system was an operating system kernel. The planned kernel, called HURD (herd of GNU's), was to be a collection of servers which ran on top of the Carnegie Mellon University micro kernel, MACH. This part of the system has been the hardest to complete and after many years of work by a large number of people, HURD is still not reliable.

In 1991, Linus Torvalds developed a Unix-compatible kernel which he called Linux. In 1992, combining the Linux kernel and the GNU system resulted in a complete free operating system which has grown and evolved to what is now known as Linux. All of the software parts of most Linux distributions are available under the GNU software license.

3 Open Software

Richard Stallman has been controversial in his unwillingness to relax any of the restrictions in the GNU software license. Many feel that his refusal to use phrases other than *free software* to describe his concept of software freedom hurts the overall acceptance of his ideas. Stig Hackvän argues [Hac 1999] that free software, because of the implications of its name, will never be widely accepted in mainstream corporate computing, while some other name, such as *open sources* does not have the implication that something which is free could not possibly be supported well enough to be relied upon. Eric Raymond, author of the well known paper "The Cathedral and the Bazaar" [Ray 1997] which is credited with providing the motivation for Netscape Corporation to release the source code for the next version of Netscape Communicator, has been the organizer of efforts to provide software licenses which preserve software freedom and are more likely to be generally acceptable in the computing industry. Examples of such licenses are the Netscape Public License [Net 1998], the Artistic License [Art 1998]. and the Apple Public Source License [App 1999].

Eric Raymond's widely read Cathedral and Bazaar paper was followed by "Homesteading the Noosphere" [Ra1 1997] where he discussed the property and ownership customs of the open source culture. Then, more recently, he describes open source business models using game theory and economic theory in the paper "The Magic Cauldron" [Ray 1999]. Out of these writing and research efforts, Raymond has been instrumental in establishing a set of licensing standards which are less restrictive than the GNU Public License in the area of what may be done when changes are made to someone else's program. The licensing guidelines are known under the name of Open Source.

3.1 The Open Source Definition

Open Source [Ope 1999] programs must meet the following criteria:

- *Free Redistribution* The license may not restrict any party from selling or giving away the software distribution containing programs from several different sources. The license may not require a royalty or other fee for such sale.
- *Source Code* The program must include source code, and must allow distribution in source code as well as compiled form.
- *Derived Works* The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
- *Integrity of The Author's Source Code* The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.
- *No Discrimination Against Persons or Groups* The license must not discriminate against any person or group of persons.
- *No Discrimination Against Fields of Endeavor* The license must not restrict anyone from making use of the program in a specific field of endeavor.
- *Distribution of License* The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
- *License Must Not Be Specific to a Product* The rights attached to the program must not depend on the program's being part of a particular software distribution.
- *License Must Not Contaminate Other Software* The license must not place restrictions on other software that is distributed along with the licensed software.

Open Source software licensing guidelines appear to be more widely accepted by many of the larger companies in the computing industry as they begin releasing some of their products to the open source community. Many software industry analysts argue that this would have never happened under the GNU Public License. The essential software freedoms of the GNU license are preserved under Open Source Licensing.

4 Reading to Write

Imagine a college level writing course, taught by the English Department, where students are never given reading assignments of great works of literature. Good

writing, like good programming, is partly an art. Most would agree that developing good writing skills involves reading great works of literature as well as doing lots of writing. Applying the same principle to the writing of computer programs should involve reading the code of talented programmers. Most undergraduate computer science curricula never provide such experiences, in fact, most available software contains restrictive license clauses which make access of source code an impossibility. GNU license software and various other open source licenses provide an opportunity for students to have access to large bodies of source code for study and perhaps modification. For example, students who have learned a programming language such as Scheme can benefit from reading the source code to a Scheme interpreter or compiler in a principles of programming languages course.

Similar cases for effective use of source code in a variety of courses is easily made. An operating systems course might examine the kernel source code to see implementation of scheduling algorithms. The author has used the source code for a version of the J [Hui 1992] programming language as an example of how a very skilled programmer has designed data structures and abstractions to deal with complexity in this interpreter. Students may be given assignments where they extend the features or functionality of some large program. Linux evolved out of such an experience with Tannenbaum's Minix system when Linus Torvalds decided to extend and re-write the Minix [Tan 1997] system after studying its source code in an operating systems class.

5 Software Service

The concept of professionals providing their services without compensation for the purpose of improving society, offering aid in a time of need, or to further the goals of the profession is well understood as required behavior for professionals. Students should be encouraged to participate in a GNU or Open Source project. Each project typically has a task list and the project manager is usually in need to volunteers. Contributions, even though small, are not insignificant. Most projects have a variety of needs, such as documentation writers or web masters in addition to programmers. Documentation is an excellent way of initial involvement in a project which often leads to other contributions.

6 Open Software and Our Graduates

Most undergraduate computer science departments include employment of their graduates by the computing industry as one of the possible outcomes. More often than not, such employment involves an employee contract which states that the employer reserves all rights and claims to all things of a technical nature produced by the employee during the tenure of employment. Whereas work on open software projects requires that all such work be made freely available in source form.

Does Open Software provide viable employment or business opportunities for our graduates? A number of companies have announced active participation in Open Software projects. These companies are employing people to make contributions to these projects. Some examples include the release by Netscape [Net 1998] of the code for Netscape Communicator program. Apple Computer [App 1999] has released the source code for its new server operating system under its open source code license. Silicon Graphics [SGI 1999, SGi 1999] has announced that it has released the code to its GLX system under the GNU license and also announced that they had embarked on a program to replace their proprietary Unix implementation, Irix, with Linux. Sun Microsystems announced [WSJ 1999] the release of the source code to Solaris under a license which is similar to the GNU license.

Computer systems manufacturers, such as IBM, Dell, and Compaq, have all announced the availability of computer systems with Linux pre-installed. Linux support programs are available from these vendors as well as independent support firms.

The use of Open Source systems, such as Linux, is becoming rather pervasive throughout the computing industry, so much so that it is relatively easy for computer science graduates to find an employer who will hire them to work on Open Source projects rather than the more traditional proprietary projects.

As an example of the kind of commitment large companies are making, Robert LeBlanc, the executive responsible for IBM's Linux efforts, said in a Linux World [Bar 1999] interview

"The only message we want to get across is that we are very serious about Linux. We see Linux as being a very important platform in the industry. We certainly believe that we bring some level of credibility to Linux, especially in commercial accounts.

We have ported key software to Linux. We support Linux on all of hardware. We just announced Thinkpad support for Linux, something that our customers have been yammering at us about for awhile. We've shown Linux running on RS/6000. We just announced that we are going to support Linux APIs and ABIs on top of our AIX operating system to allow people to move Linux applications over to AIX, and in the future, Monterey, which is the convergence of all the Unix [efforts at IBM]."

7 Open Software and Computing Laboratories

Trinity University has used Linux based systems for several years with excellent results. Linux was first used in mission critical server applications such as departmental Web servers and NFS file servers. The choice was made, not because of cost, but rather, because of our desire for reliable and secure server systems. Over a period of three years, these systems have experienced only a few hours of unscheduled down time. The NFS server functioned in a multi-platform environment providing home directories to SGI and HP Unix systems as well as to PowerPC and Intel based Linux systems. Recently, a 23 machine laboratory

of HP 712 workstations were replaced with 23 Pentium III machines running our own variant of RedHat Linux, version 6.0. These machines are connected to the Internet through a Cisco 3524 Ethernet switch so that each lab machine and the file server have a dedicated 100MB Ethernet line. The switch improves the performance of these machines when run as a 23 node parallel processing system. A dual processor Pentium III machine supplies an NIS database to all departmental Unix machines as well as home directories via NFS and Windows 9X/NT access to home directories via Samba. This machine also functions as a backup departmental Web server. A second lab of 22 Intel based Linux machines has been constructed, and, together with about 10 other Unix machines, all are being served by the single NFS file server.

The Computer Science Department maintains their own set of local changes to the RedHat distribution of Linux. These changes include security modifications, updates and installed software packages which are used in a variety of courses. We make this distribution available to students on a department FTP server so that students can install and use on their own machines the same software environment which is available in the Computer Science Department labs. Since the software bears the GNU license, this may be done at no expense to the students.

8 Conclusions

Michael Jensen [Jen 1999] argues that higher education, being not for-profit, should question its relationship with the commercial software industry. He writes "We need to decide what kind of relationship academe should have with the tools that underpin its knowledge bases – that of a huge corporate customer that goes to private industry for software, or of a supporter and underwriter of open and free software tools that serve our needs". Peter Linkins remarked, during a meeting at the Online Computer Library Center in Ohio [Jen 1999] : "A for-profit's mission is to create as much value for its stockholders as possible, within the constraints of society. The non-profits' mission is to create as much value for society as possible, within the constraints of its money." Jensen notes [Jen 1999]: "Finally, we should remember that research and scholarship are fundamentally open-source enterprises. The research on which scholarship builds is always cited; the methodology of any study is explained as a repeatable framework; and theoretical preumptions are made clear as part of any published argument. Those principles generally lead authors to be careful, and help other scholars to test an author's conclusions."

Software freedom has the potential of re-kindling some of the excitement and enthusiasm experienced in the early history of computing. Free and open sharing of software ideas and their expression is entirely consistent with the academic freedom of sharing of ideas which all of us within academia so highly cherish. Students benefit from the reading of source code of real programs written by great programmers and designers. They can see first hand the abstractions and data structures which have been designed to deal with the complexity of the

program.

By using Open Software in our courses, we can show the complexity of real programs and instrument them to produce working models of the theory and concepts we teach.

Our students can benefit from the contributions, no matter how small, they make to an Open Source or GNU project. Such service adds quality to their professional life just as the service performed by professionals in other fields.

References

[App 1999] Apple Computer, "Open Source Projects at Apple" http://www.publicsource.apple.com/apsl/, 1999.

[Art 1998] The Artistic License, http://language.perl.com/misc/Artistic.html, 1998

[Bar 1999] Barr, Joe, "IBM'S Secret Summit", Linux World, http://www.linuxworld.com/linuxworld/lw-1999-09/lw-09-ibm_2.html, September 1999.

[DiB 1999] *OPENSOURCES, Voices from the Open Source Revolution*, Edited by Chris DiBona, Sam Ockman and Mark Stone, O'Reilly & Associates, Sebastapol, California, 1999.

[HAC 1999] Hackvän, Stig, "Reverse-engineering the GNU Public Virus", Linux World, http://www.linuxworld.com/linuxworld/lw-1999-09/lw-09-gnu.html?09-21, 1999.

[Hac 1999] Hackvän, Stig, "Radical Software Environmentalism", Linux World, http://www.linuxworld.com/linuxworld/lw-1999-09/lw-09-gnusidebar.html?09-21, 1999.

[Hui 1992] Hui, Roger, K., *An Implementation of J*, Iverson Software, Toronto, Ontario, 1992

[Jen 1999] Jensen, Michael, "Information Technology at a Crossrads: Open Source Computer Programming", The Chronicle of Higher Education, October 29, 1999.

[Lin 1999] Linux Magazine, Interview with Richard M. Stallman, July, 1999.

[Net 1998] Netscape, Mozilla & Netscape Public Licenses, http://www.mozilla.org/MPL/, January 1998.

[Ope 1999] "The Open Source Definition", http://www.kr.debian.org/OpenSource/osd.html, 1999.

[Ray 1997] Raymond, Eric, "The Cathederal and the Bazaar", http://www.tuxedo.org/~esr/writings/cathedral-bazaar/, 1997

[Ra1 1997] Raymond, Eric, "Homesteading the Noosphere", http://www.tuxedo.org/~esr/writings/homesteading/, 1997

[Ray 1999] Raymond, Eric, "The Magic Cauldron", http://www.tuxedo.org/~esr/writings/magic-cauldron/, 1999

[SGI 1999] Silicon Graphics "Linux", http://www.sgi.com/developers/technology/linux/, 1999.

[SGi 1999] Silicon Graphics "SGI Linux Environment 1.0 with RedHat Linux 6.0", http://www.sgi.com/software/linux/index.html, 1999.

[Tan 1997] Tannenbaum, Andrew, S., and Woodhull, Albert, S., *Operating Systems: Design and Implementation*, 2nd Edition, Prentice Hall, Upper Saddle River, New Jersey, 1997.

[WSJ 1999] Hamilton, David, P., "Sun Microsystems, to Parallel Success of Linux, Makes Solaris Code Available", Wall Street Journal, Page B6, October 1, 1999.