Free Software in Engineering Education

Andrew Sterian
Grand Valley State University

Abstract

Software that supports engineering design and development is an important part of engineering education as a medium for teaching fundamental concepts. In several fields, an increasing amount of engineering practice is becoming software based, for example digital system design using VHDL or mechanical design using finite-element modelling. The cost of incorporating this software into engineering instruction may be prohibitive. Software companies are turning to subscription models to ensure yearly revenue, and operating system dependencies can create a cycle of forced upgrades that further increase cost. Also, students are frequently burdened by the software since licensing and cost restrictions may force them to use software in prescribed locations on campus rather than obtaining personal copies of the software and using it at home. The increasing popularity and maturity of free software can often be a strong alternative to this situation. In this paper we consider the benefits and drawbacks of free software for engineering education. Issues such as support, reliability, control and customization, portability, and software quality are discussed. We also provide some specific examples of free software for use in engineering education.

1. Introduction

As more and more software is used for engineering instruction, we are becomingly increasingly apprehensive about this software's costs and constraints on the students. For example, consider the popular scientific computation software, MATLAB. This is very useful software that supports a wide variety of courses across many disciplines of engineering, and many institutions have purchased this program either for use in a lab or as a site license. Anyone who has done so is aware of the associated high cost. In addition, if the software is licensed for a specific number of seats, students must work at a specific location on campus. Even if the software is site licensed to the entire institution, it is very cumbersome (if not impossible for some students) to use this software off campus. Students may purchase a limited version of the program for personal use (at their own expense) but the limits of the program may conflict with the needs of the course.

If MATLAB were the only program that incurred costs and constraints, it would perhaps not be a serious problem. As more and more software with similar costs and constraints becomes a part of an engineer's education, the problems compound. For non-traditional students especially, the flexibility they need in their time and location schedules is frustrated by the constraints of restricted software.

We propose that engineering departments consider replacing some of their commercial educational software with free alternatives. While this will clearly reduce costs, perhaps even more important is that it reduces constraints. Students are free to obtain and use the full version of the software at any time and in any location. There are other benefits of free software, discussed

below, but we propose that the above two issues, no cost and freedom to work anywhere, are the most significant.

The unfortunate reality, of course, is that the majority of engineering software is not free. Even for an available free software package, the differences between it and its commercial counterpart in functionality or ease of use can often be large enough to dispraise it. For some programs, the commercial offerings are the only viable alternatives.

2. Free software licenses

The term "free software" actually refers to a variety of different licensing schemes. They share in common the fact that the software costs nothing. The licensing schemes differ in whether or not source code is provided, and along more subtle dimensions regarding how the source code may be treated.

2.1 Free to use software

Software that is simply "free to use" does not have source code available. This type of software is usually a trial or limited version of a full-featured program, sometimes only available in this form for academic use.

2.2 Open Source software

Software that is formally classified as "Open Source" must meet various criteria that allow it to carry this distinction[1]. First and foremost, the source code to the software must be available. Additional components of the license prohibit discrimination against persons, groups, or specific domains of application, and provide the right to create derivative works or to modify the software.

2.3 GPL software

The oldest license to distinguish free software is the GNU Public License, or GPL[2]. Software that is distributed under the GPL not only assures that its source code is available, but it also introduces restrictions that prevent this assurance from being abrogated. The GPL is an actual license, whereas the Open Source definition is a standard that describes the minimum requirements of any software license for its associated software to be considered Open Source.

2.4 Other free licenses

Recently a proliferation of GPL-type licenses have been introduced that introduce variations in the details of how the software may be used, distributed, or modified. For example, the MPL (Mozilla Public License) which arose from the recent source code release of the Netscape browser, the QPL (Qt Public License), the IBM Public License, and the Sun Internet Standards Source License. The Open Source web page provides a listing of a wide variety of licenses that meet the Open Source definition[4].

2.5 Public domain software

Software that is available without license may be placed in the public domain by its author (thereby giving up copyright). There are no restrictions or conditions on using this type of software. Authors of significant software programs generally do not place their software in the public domain as it allows a commercial company to sell a non-free version of this software for profit with no compensation or recognition to the author.

3. Comparison of free and non-free software

As we discussed above, the two most obvious benefits of free software are no costs and no constraints. There are a variety of other issues that must be considered, however. We present these issues below.

3.1 Control over functionality

With source code available, software lends itself to customization, extensions to functionality, and the potential to find and fix bugs. Commercial software without source code affords none of these benefits. This, of course, assumes that someone familiar with the source code and the programming language is available for effecting the necessary changes.

3.2 Ease of use and documentation

Commercial software authors often have significant resources to develop and refine sophisticated user interfaces. Free software, as a rule, focuses on functionality above user interface. Using free software, then, may require a greater effort on behalf of faculty for making the software accessible and easy to use.

The same principles apply to installation and documentation. Single-click installation programs, common for commercial software, are the exception for free software, and the latter often have dependencies that must first be located and installed (e.g., external libraries, drivers, etc.) Similarly, the documentation that accompanies free software is frequently terse and written as a reference rather than a tutorial. Once again, the degree of faculty involvement is higher in order to overcome these deficiencies.

There are exceptions to the above, of course. On average, however, commercial software is easier to install, has a more polished user interface, and has ample tutorial and example material available.

3.3 Support

The support available for free software is surprisingly good, but it does not take the form of a toll-free number. Free software is generally crafted by authors that take a personal pride in their work, and they are usually quite interested in offering software that is of very high quality. Often, sending e-mail to the authors of free software detailing a newfound bug or other program problem elicits a quick response, and may even come with a fix attached. Free software programs often

have USENET newsgroups, mailing lists, or web-based discussion forums established which provide support through other users or the program authors themselves. One-to-one support from the program authors becomes less likely as a program's user base grows large, and the number of "noise" questions increases. In this case, the many-to-many facilities (mailing lists, newsgroups, web forums) become the main support mechanisms.

Increasingly the trend for commercial software is to sell support at additional cost, often with several levels of support metering the number of incidents or time allotted for a given price. Newsgroups and mailing lists may exist for the software, as they do for free software. Actual bug fixes may be a long time in coming, however, as it is an unfortunate truth that customers do not want to pay for bug fixes, only new features. The infamous "next version" of the software is promised to address all users' complaints, but at a price, and with many new and possibly useless (or bug-ridden) features added.

Free software programs that are "labors of love" are more likely to be abandoned than commercial programs. This can also happen to commercial programs. More common, however, is that a particular version of a commercial program is abandoned in favor of a new, major release and the support for the older version is terminated. Users who have no need for the new version are still forced to purchase it if they want continuing support or additional copies of the software. This "forced upgrade" scenario is much less likely for free software with source code provided. Even if the original author abandons the project, the source code may fall into the hands of others who are willing to take up the project. A new version of free software that is somehow undesirable does not force an upgrade since the source code is available (for support) and as many copies can be obtained as necessary, since it is free.

3.4 Institutional inertia

It may be difficult to replace commercial software with free software when it is used in several courses and several faculty members are affected. This requires either an extremely high degree of compatibility between the two alternatives (very unlikely, especially as new versions are released) or the agreement among all faculty to proceed down the free software path.

3.5 Compatibility with existing software

When free software must exchange data with commercial programs, the issue of data compatibility arises. For example, the free StarOffice suite of office programs is intended to be compatible with Microsoft Word, but the level of compatibility is not 100%. If some are using StarOffice while others are using Word, there will certainly come a time when these incompatibilities will cause difficulty. Unless an entire, isolated infrastructure of free software is in place, a single free software program in the midst of commercial programs is a concern (when data must be exchanged).

3.6 Portability to alternative platforms

Free software is often available for several platforms (hardware and operating system combinations). Commercial vendors must justify development costs for platforms that are not profitable.

If a platform is seen as not profitable, no version for that platform is released. When source code is available, however, a dedicated group of users working on a particular platform often arises and provides a port of the software to that platform. For this reason, and also perhaps because the user interface (often neglected in free software) is a large part of the problem in portability, free software tends to be available on more platforms than commercial software. Once again, the constraints on students are lessened if they are free to use their hardware and operating system platform of choice.

3.7 Training for industry

Sometimes, a commercial package is chosen since it is the program most expected by industry for which the students are being trained. This is a poor reason for choosing a program. Engineering students must receive an education, a foundation that will last throughout their career and this cannot be dependent upon the offering of a particular vendor. A free software alternative that conveys the same concepts or framework is a perfectly justifiable program. An employer who demands that students be trained on program X even through program Y teaches the same concepts indicates a lack of investment in employees. Surely the employer would let the student invest some time in transferring the concepts learned from program Y to the specific keystrokes and menu options required in program X.

4. Examples

We briefly present several free software programs that are very viable alternatives to commercial programs. A detailed comparison between the free vs. non-free alternatives is at best a moving target; please contact the author for empirical observations.

4.1 Scilab

Returning to the example of MATLAB software from the Introduction section, we have chosen the free program Scilab[3] to support our senior-level EE/CE course on Digital Signal Processing. This program is available under a GPL-style license. Source code is available, it is portable to both Windows and UNIX-type platforms, ample documentation is available (including a textbook), and a USENET support news group is active (and monitored by the program authors). The Scilab program is not a direct drop-in replacement for MATLAB, but both programs are based on the same matrix-oriented computational environment.

Of the two, MATLAB is clearly more mature, diverse, and well-represented in industry. For industrial applications, the choice of MATLAB vs. Scilab would require a fair amount of comparison. For instructional use, however, Scilab is completely adequate. There are several aspects of Scilab, in fact, that are superior to MATLAB. For example, Scilab has a built-in polynomial data type that nicely supports some symbolic manipulation for signal processing (without requiring a MATLAB-like cumbersome symbolic toolbox extension).

Also, Scilab is not the only free MATLAB-like program available. Other programs such as Octave and R-Lab are also free and are also most likely quite adequate for instruction.

4.2 Eagle

For our EE/CE courses on digital circuit design, we have found the commercial Eagle program[5] to be a good choice. This program is free to use for academic purposes but does not come with source code. The free version of this program also places restrictions on maximum circuit board size and number of layers, but we have not found these restrictions to be severe.

Unfortunately, there is no viable program for digital circuit design (encompassing schematic capture, board layout, and autorouting) that is truly free. The gEDA program[9] that is currently under development shows promise, however, and may be a viable competitor over time.

4.3 StarOffice

As mentioned above, the StarOffice suite of office applications[6] is available for free and is a very robust package. Included applications are a word processor, spreadsheet, presentation graphics, and database program, among others. StarOffice is licensed under the GPL, hence the source code is available. As students make heavy use of office programs for report writing, presentations, and spreadsheet computations, a free office suite can realize great savings in costs.

4.4 SampLin

SampLin[10] is a free package for data acquisition, process control, and visualization. This Linux-only program may be an alternative to the popular National Instruments LabView data acquisition/control environment (as an example). SampLin includes support for popular device interfaces such as serial and GPIB, a scripting language, and network-based data acquisition. SampLin is a nascent program and cannot compete with the mature LabView software, but it may be more than adequate for basic data acquisition and control.

4.5 LPC

Linux Programmable Controller (LPC)[8] is a programmable logic controller (PLC) emulator. Manufacturing Engineering courses that rely on PLC's for laboratory experience can be expensive due to equipment and software cost. The free LPC software can minimize the dependency on equipment by providing an alternative experimentation platform, plus LPC can even act as a limited PLC itself with hardware support for several devices.

4.6 GNU/Linux

The GNU/Linux operating system is now a well-known free alternative to commercial operating systems. It is not an engineering application, but it should be considered as a viable candidate for hosting engineering applications rather than the more common Windows operating system. For example, all of the applications listed above in this section will run on the GNU/Linux operating system. Other free Unix-based operating systems are also available (e.g., OpenBSD, FreeBSD).

Using GNU/Linux together with Windows-based applications is also a possibility with emulators

such as WINE[12] (a free Windows emulator for Unix systems) or Win4Lin[13] (a commercial emulator) and virtual computer software such as VMWare[14], which allows running multiple operating systems simultaneously. These programs may be useful in initiating a transition from a Windows environment to the free GNU/Linux environment.

4.7 Other applications

A review of existing free programs (see, for example, the category of Scientific Applications on the web directory Freshmeat.net[7]) reveals free programs for finite element analysis (SLFFEA, freefem), discrete event simulation (OMNeT++), languages for statistical computing (R, PSPP), SPICE simulators (Spice Opus, ngspice), and many others.

Bibliography

1. URL: <http://www.opensource.org>, The Open Source web site
2. URL: <http://www.fsf.org/copyleft/gpl.html>, The GNU General Public License
3. URL: <http://www-rocq.inria.fr/scilab>, Scilab, a computational algebra program
4. URL: <http://www.opensource.org/licenses>, A listing of software licenses that meet the Open Source standard
5. URL: <http://www.cadsoft.de>, CadSoft, authors of the Eagle electronics CAD program
6. URL: <http://www.sun.com/products-n-solutions/software/prodapps>, StarOffice, a free office application suite
7. URL: <http://freshmeat.net/browse/97>, The Scientific/Engineering section of Freshmeat.net, a current software directory
8. URL:<http://claymore.engineer.gvsu.edu/lpcd>, Linux Programmable Controller, a PLC emulator
9. URL:<http://www.geda.seul.org>, gEDA, a suite of free software tools for electronics CAD
10. URL:<http://www.iaee.tuwien.ac.at/sensor/samplin>, SampLin, free data acquisition/process control software
11. URL:<http://www.linux.com>, GNU/Linux, a free operating system and kernel
12. URL:<http://www.winehq.com>, WINE, a free Unix-based emulator for Windows applications
13. URL:<http://www.win4lin.com>, Win4Lin, a commercial Windows emulator for Linux
14. URL:<http://www.vmware.com>, VMware, a commercial virtual computer environment for running multiple operating systems simultaneously

ANDREW STERIAN
Andrew Sterian is currently an Assistant Professor in the Padnos School of Engineering at Grand Valley State University. He received his B.A.Sc. in Electrical Engineering from the University of Waterloo, Canada and the M.S.E. and Ph.D. in Electrical Engineering from the University of Michigan, Ann Arbor. He has taught courses in signal processing, digital systems, and microcontrollers.