Understanding Open Source Software

***B. Wayne Walters*[1]**

Abstract – Experience in the classroom has shown that students have a limited understanding of Open Source software. Many students view Open Source software as just a free alternative to proprietary software. Students are always interested in free, whether it's free software or free pizza. At other times, students' understanding of Open Source is expressed simply as a dislike of Microsoft products.

Educators need to provide a better understanding of the importance of these software licenses to their students. Open Source software is gaining popularity. An Open Source product, OpenOffice, is replacing MS Office in some schools districts around the US. With tight school budgets, superintendents are realizing the cost savings of using the OpenOffice products on their campuses.

Lectures on Open Software are being introduced into the IT Project Management course at the University of Southern Mississippi. These lectures have focused on the economics of using Open Source for systems, tools, and applications.

Keywords: Open Source Software, Open Technology Development, GPL licensing

OPEN SOURCE VERSUS PROPRIETARY SOFTWARE

There seems to be a lot of misinformation and mystery surrounding the topic of Open Source (OS) software. This lack of a clear understanding is also quite prevalent in professional ranks, so it is no wonder that students have misconceptions regarding what OS is and how best to take advantage of its potential. Many students voice support of Open Source software simply because they dislike Microsoft's or some other proprietary vendors' products. This dislike is often fostered by peers or discussions with professionals in the computer field. Computer professionals are likely polarized into 2 camps – Microsoft and non-Microsoft. This can make it difficult for novices and professionals alike to create an informed opinion since a broad spectrum of people have limited knowledge or have biased opinions about Open Source software and how it can be best utilized.

Proprietary Software

The most widely used software products are proprietary ones. There are proprietary software solutions that are computer systems and tools, such as Microsoft XP, and there are software solutions that are applications specific, such as, an integrated, enterprise program to manage the recordkeeping for an organization. With proprietary solutions, users can be controlled by their vendors. That is, users can become dependent upon a product's features and interfaces and can have a difficult time changing to a new software platform. Startup costs associated with installing new software can also have an impact on the feasibility of changing software vendors.

With application software, the problem can, in some situations, be more critical. For example, if a university has spent more than a million dollars purchasing hardware, software, training employees, and hiring new employees for their new application, they are not likely to want to change vendors. If, later, they find that they are unhappy with their vendor or want new features that the vendor is not willing to provide, they may have a problem. The cost of new resources, data conversions, and the resulting impact on operations can be tremendous and may create more

[1] The University of Southern Mississippi, School of Computing, 118 College Drive, Box 5106, Hattiesburg, MS 39406 wayne.walters@usm.edu

problems than they are willing to endure. Conversions to a new package could take more than a year to install after a decision to change vendors is made. Vendors know this and often take advantage of vulnerable clients. This is referred to as vendor lock-in. When Open Source software is designed and created with extensibility in mind, it is possible for the software to evolve as needs change since the client would have source programs available for modification.

Open Source Software

The Information Technology program of the School of Computing at the University of Southern Mississippi (USM) is trying to change some of its student's misconceptions about OS. Lectures have been incorporated in a Project Management for IT course to give students a better understanding of what OS is and how it may be licensed. The future careers of engineering and computer professionals alike may require them to deploy software solutions for their employers. They need to understand the software options available to them and how these options can effect their software deployments in the future.

Open Source software is often perceived as a limited set of free or low cost alternatives to proprietary software packages. Based upon ones perspective, ones view of OS may be limited to such packages as Linux, Apache, MySQL, Perl, OpenOffice and a few other solutions. Many other software packages are available as applications. The website SourceForge.org [9] offers a repository of thousands of Open Source products. Additionally, SourceForge.org hosts many developmental resources for creating and managing OS software projects.

The developers of OS products do not usually establish marketing agendas for the products that they create. They are seldom a part of a "for profit" organization. Many of these developers are somewhat altruistic and donate their services to provide Open Source software. Marketing companies have evolved for open source distribution and enhancement and do so from a marketing standpoint of support and training of users of their supported products. These companies have little interest in whether one has the latest version of software or not, only that the clients have the software that they are supporting. An example of this model is RedHat Linux. RedHat Linux was formed for the purpose of marketing the Linux software. Although they didn't write the software, they modify and add tools to their marketed products.

Open Source Software Definitions

The definition of OS offered by Wikipedia [10] is "Open Source describes practices in production and development that promote access to the end product's source materials--typically, their source code". The second point is the one that best describes OS, that is, "promote access to the end product's … source code". This is where the meaning of the name is derived. The other point, the "production and development' of software describes how the software is developed. Developers of OS are quick to link the development of the product and the open availability of the source together when describing it. The author feels that this should be part of a second definition, Open Technology Definition, which is discussed later.

Open Technology Development

In the Wikipedia definition of Open Source offered above, there were two main points. One of the points addressed open access to the source code as part of the definition of Open Source software. The other point spoke of production and development practices. Within the Open Source community there are many who have liberal concepts that believe that the open exchange of ideas, i.e., source code, in the commercial world should be practiced as it in the academic arena. The concept of access to source code is evident in any definition of open source but a definition of how the software is developed stretches that definition. The author prefers to separate this definition into two components – Open Source Software and Open Technology Development (OTD). The OTD addresses the open nature of development practices. The OS community prefers to rigidly connect these two methodologies. The development of early OS software grew out of very open and public development practices, most notably with the GNU Project [4] and the Linux development efforts. This development, as practiced, allowed for contributors' efforts to be open to review by anyone who wanted involvement in the project. This paradigm provided a higher quality product because of its frequent open reviews by highly skilled professionals. This methodology is generally credited with the resulting robust and secure Linux operating system.

For the definition of Open Technology Development, the term as described by the Department of Defense (DOD) in its *Open Technology Development Roadmap* [Lucas, 2] is used. The OTD methodology incorporates the following tenets: 1) Open Standards and Interfaces, 2) Open Source Software Designs, 3) Collaborative and Distributed online tools, and 4) Technological Agility

<u>Open Standards and Interfaces</u>

As software becomes more networked, the need for modularity and extensibility grows. If standards and interface definitions are open to the development community of interest, then new solutions can be provided by many suppliers. This can allow later tailoring of components to meet differing needs. This component tailoring or modification would be difficult without open standards and interfaces. An example of these types of standards is the Global Justice XML Data Model (GJXDM) created by the Department of Justice [6] and Georgia Tech University [5] and used for law enforcement records systems.

<u>Open Software Designs</u>

Open Software Design makes the design of a system open and visible to the community of interest. The success of this design methodology lies in the process. Open Design, like open development, relies on many diverse, skilled "eyes" to review the plan and provide constructive criticism. The effectiveness of this open exchange of ideas has surprised many. The community of interest that is created usually is made up of volunteers who want to be involved. They often have not met face-to-face and are not bound by justifying their positions with an employer.

<u>Collaborative and Distributed Online Tools</u>

Open Source software projects have nurtured communities of interest with people from many backgrounds, in different geographic locations, and possessing many diverse skills who work together to produce a solution. Since many of these developers never meet face-to-face, they have created many tools for communicating. These tools, enhanced by the Internet, provide facilities for online collaboration, error logging, and configuration management. These include products, which are likely the result of OS development themselves, like Bugzilla, CVS, RSS, and GCC tools.

<u>Technological Agility</u>

The Technological Agility factor recognizes the fast pace of technology and the need for understanding the evolving of leading-edge concepts. To utilize this agility a person of vision is needed. This is a critical step for developing an outstanding Open Source product.

Open Source Software Gains Popularity

Open Source software's history began with various university projects and has some links to early shareware and freeware software programs. Some of the first software products to carry the label, Open Source, were tools and derivatives of the UNIX operating system. The Linux operating system is the best known of the OS programs. Within the OS software community, the most important aspect of Linux is that the source code is freely available to anyone. Also, if there is a feature that is desired by you or the community of OS/Linux users, you can freely add that feature and use it or submit it to the Linux repository for distribution to others.

Open Source software is sometimes referred to as free software. In some cases it can be obtained at no cost or low cost. Unfortunately this is how many people think of Open Source. Although many OS products are free, many are available with a choice of two license agreements. One license may be for a free or low cost plan and the other may be fee-based. The fee-based plan usually provides a wider range of support options, such as, software support and training. One must be very careful when using OS to ensure that the licensing agreement that is connected with the products is understood. When obtaining software products, there are other reasons to consider OS than just a pricing model. Open Source software has a reputation of solving many critical software issues, such as vendor lock-in, extensibility, security, interface standards, and even performance [Winslow, 3].

Licensing Models

The Open Source Initiative (OSI) has copies of many of the OS license models that are available. The OSI recognized licenses are based upon their Open Source Definition (OSD). The OSD is a framework for OS licenses, not a requirement. The OSD states that OS means more than freely distributed source code. They state that it should include the following topics [7].

1. Free Distribution
2. Source Code
3. Derived Works
4. Integrity of Author's Source Code
5. No Discrimination Against Persons or Groups
6. No Discrimination Against Fields of Endeavor
7. Distribution of License
8. License Must Not Be Specific to a Product
9. License Must Not Restrict Other Software
10. License Must Be Technology-Neutral

Some of the points in the OSD have created problems for some OS developers. For example, a product called OpenSSL uses encryption methods that are protected by US export restrictions. The development of this product was forced off-shore because of US regulations. It could not be released honoring the OSD within the US because it must discriminate against groups and fields of endeavors as required by export restrictions.

Of course, Open Source licenses always allow for the distribution of the source code of the software but how the distribution is handled varies significantly with some of the major license types. Some licenses require that modification to the source be returned to the author before redistribution and others don't. Below are just a few of the popular licenses that have been created [Fink, 1].

- The BSD-style License allows free use and redistribution of binaries and code without "check-ins" from the public.

- The CopyLeft, Linux-style License (or GPL) requires that all derivative works in turn must also be GPL code. The "CopyLeft" principle that was included means that you can't modify the source and make a profit without returning those modifications.

- Lesser GPL License was created because the GPL was interpreted by many people to mean that anything that touched a GPL licensed product was part of the GPL. In the case of an operating system, did this mean that any third-party libraries were also GPL? The Lesser GPL was to solve this problem.

There are many other OS licenses available that could be applied to new products. If one is interested in developing Open Source software and releasing it to the public, careful consideration must be given to the complex legal issue of licensing

Future Implications of Open Source Developments

In the lectures to students about OS, it is important to note that the discussions should not be just about what has happened in the past. That is, when students are asked to write a short paper on what Open Source software is, invariably their discussion will be about Linux. In the course it is emphasized that there is an OS future and much of that future may be in the area of applications or enterprise software. To stimulate their thoughts, possible models for developing applications are discussed.

The next movement in Open Source development may be in local government and other public applications projects. Systems such as university recordkeeping would make an excellent project. This is the large integrated system (Admissions, Registration, Parking Management, Alumni, Sponsored Programs, Athletics, etc.) that university faculty and staff deal with on a daily basis. Let's look at a possible scenario. Let us assume that the U. S. Department of Education (DOE) would fund a project to define a universal data model for educational records systems similar to the DOD's Global Justice XLM Data Model for law enforcement mentioned earlier. Further let's assume a DOE grant to develop the universal records system for colleges and universities was offered. Using the DOE data model, an interesting system could evolve. Its might have the following characteristics:

- Paid for once (DOE grant), given away to all free/low-cost.
- Source code and binaries provided to any interested party.
    - Colleges/Universities.
    - Software Integrators.
- Users would be free to replace or modify any module.
    - Re-write modules such as, alumni, parking management, or registration.
    - Re-written modules placed in a repository for other users to access.
- A support industry would be created around the product.
- New standards based system could evolve over time as hardware and software changes.

This model has been patterned after a system for justice and law enforcement known as the Public open source Safety Environment (PosSE) designed by USM and the Open Source Software Institute (OSSI) [8]. It is standards based and uses the GJXDM. The Jail Management module has been developed with this model and the subsequent modules are waiting build-out.

The future of the Open Source paradigm is still uncertain, but it offers many interesting prospects. When deciding between OS and Proprietary systems, there is no universal good or bad choice. There is only the right choice for a given situation. Those who make these decisions need as much unbiased information as possible to make their decisions when either procuring or developing software packages. It is the purpose of the Open Source lectures in the School of Computing to expand the understanding of Open Source software development and licensing practices by students so that they can make these informed decisions in the future.

REFERENCES

[1] Fink, Martin, *The Business and Economics of Linux and Open Source*, Prentice Hall, NJ, 2003, pp 42-48.
[2] Lucas, Mark and John Scott, *Open Technology Roadmap*, http://www.lulu.com, 2005, pp 14-16.
[3] Winslow, Maria, *The Practical Manager's Guide to Open Source*, Open Source Migrations Publishers, 2004, pg, 5.
[4] http://www.gnu.org/gnu/thegnuproject.html
[5] http://justicexml.gtri.gatech.edu/workshop/Day3/22_IndustryPerspective.pdf
[6] http://it.ojp.gov/topic.jsp?topic_id=228
[7] http://www.opensource.org/docs/definition.php
[8] http://www.oss-institute.org
[9] http://www.SourceForce.org
[10] http://en.wikipedia.org/wiki/Open_source

B. Wayne Walters

Mr. Walters holds a Masters of Science degree in Computer Science from The University of Arizona. Mr. Walters' career includes academic and professional appointments. He has experience working at the Johnson Space Center, with secure government contractors, and directing law enforcement database applications development. His interests are in open source software deployment, programming, applications design, and project management.