The Open Source Software Paradigm

B. Wayne Walters, (E-mail: wayne.walters@usm.edu), University of Southern Mississippi

ABSTRACT

A lot misinformation and mystery surrounds the topic of Open Source software. The conceptual misunderstanding of software is fostered, in some respects, by a lack of understanding within the professional ranks of the computer field. There are many different views of Open Source software today, based upon ones perspective. After about 20 years of evolution, the position of Open Source in our economy still has not coalesced. Few seems to understand whether Open Source is free software for running a computer (i.e., Linux) or just a way to obtain some software products without giving Microsoft ones money. "There is one thing stronger than all the armies in the world and that is an idea whose time has come" - Victor Hugo

Many view Open Source (OS) software as just a free alternative to proprietary software. The term does not mean that there is no cost associated with its procurement, although some vendors do provide free versions. The word "free" usually means the freedom to copy, modify, or distribute the software. One must be very careful when applying this broad, general meaning to Open Source software. There are many different Open Source licenses in use and it is imperative that users acquaint themselves with the legal ramifications of dealing with intellectual property rights.

Open Source software should be viewed as a viable alternative in many software selection processes. Recently, the news has portrayed the report of India's Kerala state moving to Linux and OpenOffice for its 2,724 high schools. Likewise, OpenOffice is replacing Microsoft Office in many schools districts around the US. With tight school budgets, school superintendents are realizing significant cost savings by using the OpenOffice products for their campuses. Products such as OpenOffice may not be 100% compatible with their current proprietary counterpart, but for many applications, compatibility is not the issue. The price and the software license management of proprietary packages may be overwhelming.

Open Source software represents a new way of dealing with intellectual property. Software is not a tangible product that can be placed on a shelf or stored in a warehouse. Software is not purchased, rather the right to use the software is granted by a license agreement. Some businesses have tried to deal with software as though it was a purchased commodity. That is, companies might tell their purchasing department to "buy" 10 copies of Microsoft XP Operating System and utilize them until the next version of the Operating System is released by Microsoft. Most businesses are learning that these licenses must be managed. But businesses have not yet obtained a clear understanding of OS software. This lack of a clear understanding of OS is still quite prevalent with many professionals in the field, so it is no wonder that there are misconceptions regarding what Open Source is and how best to take advantage of its potential. A case can be made that software belongs more in the service economy that the retail economy.

Many have voiced support of Open Source software simply because their dislike of Microsoft or some other proprietary vendors' products. This dislike is often fostered by peers or discussions with other professionals. Computer personnel are often polarized into 2 camps – Microsoft and non-Microsoft. This can make it difficult for business professionals to create an informed opinion regarding the types of software available. Which ever camp one is in, they are very likely to oppose or speak unfavorably about the others' position. Discussion about Open Source software have become very polarized and are like the arguments of the 1960's about whether Fords or Chevrolets were the best brands. These arguments are often based on ill-formed, biased opinions, not understandings.

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. The website SourceForge.org [8] offers a repository of thousands of Open Source products. Additionally, SourceForge.org hosts many developmental resources for creating and managing Open Source software projects.

The developers of OS products are usually not concerned about establishing marketing agendas for the products that they create. They are seldom a part of a "work for pay" organization. The companies which become involved in *open source* distribution and enhancement often do so from a marketing standpoint of support and training users of their products. Since their revenue streams are derived from support and training, they have little interest in whether one has the latest version of software, only that the clients have the software that they are supporting. An example of this model is RedHat Linux. The RedHat company 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 product.

DEFINING OPEN SOURCE SOFTWARE

Attempts to define Open Source software have in some way confused the matter. Simply stated, the author defines OS software as software that provides the source code of the developed software product to a purchaser when a user license and the product are obtained. Some of the confusion lies in attempts to extend this definition to meet other goals. For example, the Open Source Initiative (OSI) has created an Open Source Definition (OSD) [4]. The OSD is intended to provide a framework for creating OS licenses. The OSD states that OS means more than freely distributed source code. They state that it should include statements about the following points: 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, and 10) License Must Be Technology-Neutral. This may be a fine and worthy framework but it should not be used to define the Open Source product.

There are other factors that are often used in defining OS software. Within the Open Source community, most have liberal concepts that believe that the open exchange of ideas in the commercial world should be practiced as it in the academic arena. Most of those who are actively developing OS software believe that the definition must include openness in the development process for the software to be called Open Source. 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 OS community prefers to rigidly connect these two definitions into one definition of Open Source.

The author prefers to separate this definition into two components – Open Source Software and Open Technology Development (OTD). For the definition of Open Technology Development, the term as described by the Department of Defense (DoD) in its Open Technology Development Roadmap [Lucas, 1] is used. The OTD methodology usually involves 1) Open Standards and Interfaces, 2) Open Source Software and Designs, and 3) Collaborative and Distributed online tools. The OTD methodology addresses the open nature of development practices. Separating the definition is in no way limiting source code from freely being distributed but rather recognizes that some development projects may not be performed in an open manner. The source code can still be made freely available.

The development of early OS software grew out of very open and public development practices, most notably with the GNU Project [3] 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.

OPEN SOURCE LICENSES

Of course, Open Source licenses always allow for the distribution of the source code of the software but how the distribution is handled is dictated by the license. Some licenses require that modification to the source be returned to the author before redistribution and others don't. A few descriptions of popular Open Source license types have been listed below as examples.

- The BSD-style License allows free use and redistribution of binaries and code. While users are allowed to modify the code, the development team does NOT typically take "check-ins" from the public.
- The Apache-style License takes the BSD-style open source model and extends it by allowing check-ins to the core codebase by external parties.
- The CopyLeft, Linux-style License (or GPL) takes the Open Source license one critical step farther. Whereas BSD and Apache style software permits users to apply their own license terms to their modified code (e.g. make it commercial), the GPL license 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 to the original author.
- Lesser GPL License was created because many interpreted the GPL to mean that anything that touched a GPL licensed product fell under the umbrella 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 created to resolve this problem.

There are many other OS licenses available that could be applied to new OS 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

ECONOMICS OF OPEN SOURCE

Many believe that the Open Source paradigm is not sustainable. Yes it was good for the Linux Operating System, the MySQL database, and a few system tools, but does it have a future? Open Source software, when coupled with OTD, is a very powerful force. The quality of the products produced by this methodology is seldom questioned. Software that is of a systems nature, as opposed to application programs, tends to attract many of the best software developers. They are excited about involvement in large, complex systems that deal with the internals of computer hardware. These experts are willing to donate their time and talents to a task that produces better tools for everyone. If they are going to give their time and talents away for free, then they expect that the products they produce will be free also. This paradigm has produced many good products.

But can this model work for all software development? There are some potential problems in process. If OS software is maintained by donated workers, what happens when the workers decide that the bugs to be fixed are too trivial for them to donate their time and efforts?

OPEN SOURCE SOFTWARE APPLICATIONS

As discussed earlier, many OS products are related to systems and system tools. The question to be asked is whether enterprise applications are targets for Open Source development. Bruce Perens in his paper, "The Emerging Economic Paradigm of Open Source Software" [7], argues that there are two form of enabling technology in software: differentiating and non-differentiating. The differentiating software is software that separates you from your competition. The non-differentiating software is the software that is widely used. Examples of non-differentiating software are inventory and accounting applications. Applications that are non-differentiating are excellent candidates for Open Source because of they are widely used. If one has differentiating software, software that gives ones company a competitive edge, then one would not want it made openly available to competing companies.

Applications software development is an area where the separation of Open Source software from the development process may be needed. The chances of finding software developers that are interested in donating there time to develop a software package for managing something like a court docket would likely be very slim. But a court docket program could be developed in a closed environment but released in with an open source license. What would this model produce? For a software development firm that is profit motivated, there will always be training and support services needed. If the product has a broad based appeal, then a market can be developed. A new economic model may develop around support services for existing software packages.

PUBLIC SERVICE APPLICATIONS SOFTWARE

Some of the best examples of a new OS software model are in the public sector. A Criminal Justice and Law Enforcement system, if implemented as Open Source, would benefit the public as a whole. As this application is currently provided, there is a shortage of quality products. There is no standardization of data elements. There are little, if any, mechanisms for sharing data between agencies and judicial branches. Most noteworthy is that whenever a replacement package is needed, everything starts over; new purchase price, new equipment, new training cycles. It is out with the old and in with the new!

When applications lend themselves well to the establishment of Open Standards and Interfaces, they are a target for an Open Source product. An example of this type of standard, used in law enforcement records systems, is the Global Justice XML Data Model (JXDM) [5] created by the Department of Justice and Georgia Tech University [2]. With a system such as this, an opportunity for real cost savings in the public sector emerges. With the development of such a law enforcement package, there could be a huge savings from paying for the software once and using it nationwide many times. This could be done with a government grant. By using Open Standards and Open Software, the software would be available for anyone to customize for their specific needs. A modification of the base version could produce regionally defined versions, such as, a Mississippi version or an Arizona version. Since Open Standards were used in this example, new modules could be added or modifications could be implemented more easily. Where is the economic incentive for such a package? By defining the base system for the country to use, installation, training, modification, and support services could provide a steady stream of revenue for independent software vendors (ISVs). The MASP and PosSE [6] systems are examples of this model in use. A University Registration and Management systems is another example of this "write it once, and give it away" model.

PROPRIETARY SOFTWARE APPLICATIONS

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 a business.

There is no secret that proprietary software vendors manipulate their customers with marketing ploys that force them to pay very heavily for their software solutions. Enterprise systems, that is, those that are design to integrate everything in an organization into one big, monolithic software package, greatly restrict the options of their purchasers. The cost of implementing these solutions is tremendous. There is the cost of purchase, the cost of new hardware, the cost of new personnel, the cost of training, and the cost of support. Once an investment of this magnitude is made, it is very difficult to unseat the vendor. Enterprises are somewhat at the mercy of their vendor. Proprietary software products are notorious for not meeting the needs of their customers. Vendors are known for refusing to modify software to meet individual customer needs, providing poor support, and forcing migration to new versions at significant costs. These propriety software "games" do not provide a comfortable environment for their customers. This is one of the reasons that Open Source software has emerged as an alternative.

Another area that makes business feel very uncomfortable is the maintenance of software license. Keeping inventory of what software belongs on each computer is a major task, when performed properly. It would not be unreasonable to find a computer with 20 or more different licensed software packages. The Business Software

Alliance has raided businesses, with federal marshals, demanding documentation that the company purchased a license for every computer on site. The time and effort spent by companies to comply with onerous licensing rules is greater than most would believe. It is very easy for an employee to think that since they have licensed Windows XP, when they really have licensed Windows XP on one specific computer. That is, they cannot take XP off of a laptop and put on a desktop computer legally. Open Source would avoid that unproductive nonsense.

CONCLUDING REMARKS

In the past it was a simple situation when software developers were creating products. The products were in-house developments for their employer or a contract arrangement for a company where the one writing the contract retained all rights to the program. These licensing agreement were pretty straightforward, we own it (in-house), you own it (contract), or you can use it (software vendor). Software licensing is far more complicated today, but by obtaining a software product, such as an Open Source product, more flexibility may exist for the user of the software. With the expenses of obtaining, installing, extending, training and supporting important software packages, it is prudent for businessmen to understand the ramifications of the type of software that they use to run their businesses.

REFERENCES

1. Lucas, Mark Lucas and John Scott, Open Technology Roadmap, http://www.lulu.com, 2005, pg.
2. http://justicexml.gtri.gatech.edu/workshop/Day3/22_IndustryPerspective.pdf
3. http://www.gnu.org/gnu/thegnuproject.html
4. http://www.opensource.org/docs/definition.php
5. http://it.ojp.gov/topic.jsp?topic_id=228
6. http://www.oss-institute.org
7. http://perens.com/Articles/Economic.html
8. http://www.SourceForce.org

NOTES

NOTES