Metatron Technology Consulting's Strategic Guide to Open Source Software

Chris Travers

April 30, 2004

Abstract

IT departments are currently struggling to determine how Open Source Software can be utilized in business settings. This issue has deeper ramifications than those usually discussed, which pertain to many issues including how software is procured and managed. This paper will attempt to discuss the strategic implications for commercial and non-commercial open source software. In general, we recommend a combination of commercial and non-commercial open source software where possible, and proprietary software only where necessary.

Contents

1 An Overview of Proprietary Software

Software is protected under copyright law, which prohibits the user from copying, redistributing, or even installing the software on more than one computer without express permission from the copyright holder. This protection is designed to ensure that the right of software publishing houses, such as Microsoft or Adobe to make money selling copies of the software they develop.

As with any published product developed by a company, the majority of cost develops before the product even makes it to market. Costs such as research and development and marketing are far higher than the actual cost to actually print and transport one copy of the software. This equation leads to a tremendous economy of scale– the software same software costs much more if only 1,000 copies are sold than if 1,000,000 copies sell.

In order to be better able to spread the cost of the software around (and charge more to larger customers), software companies such as Oracle, Adobe, and Microsoft also rely on contract law to enforce additional constraints regarding the use of their software. This is accomplished using an "End User License Agreement" or EULA, which may place additional constraints on the use of a given piece of software. These constraints may include, for example, accepted usage of software including maximum numbers of connections, users, etc.

Proprietary software manufacturers include Microsoft, Oracle, and Adobe.

1.1 Sales and Support of Proprietary Software

In general, the sales of the software licenses is used to subsidize all other aspects of operations at proprietary software manufacturers. This includes support. The large software manufactures then hire dedicated staff who are responsible for supporting the software, including writing bug-fixes. In no event are the original developers contacted, as their time is usually occupied writing the next iteration of the product.

Support is also available through a wide range of other companies including independent consultants. These parties can provide support for various aspects of operation, including implementation, but are not in a position to supply bug-fixes because they have neither the right to access the source code nor redistribute modified versions of the software.

Newsgroup-based support is often available, but in my experience has been lacking due to little real interest in community support or resources.

1.2 Advantages of Proprietary Software

Historically, proprietary software has been the only option in many areas of operation. Even today, many (if not most) vertically targeted applications are proprietary. This includes medical operations software, etc.

IT managers often assume that support for proprietary software is more reliable because the software development companies are in a position to defend their name, and because support is usually subsidized by the licensing fees. In many cases, this may be true, though in most others, the perception probably doesn't actually hold up that often. However, the perception ensures the continuation of an environment where proprietary software is a "safe" bet.

Currently the IT industry is experiencing a shift toward open source software, however, use of software, particularly on mission-critical applications is generally considered by most managers to be daring, while the use of proprietary applications is considered to be safe. It should be noted, however, that most major open source programs do have vendors which will provide 24x7 support (see section 3, and are generally in a better position to offer quality support than third parties with proprietary software.

1.3 Disadvantages of Proprietary Software

When a business uses proprietary software for their operations, they are essentially responsible for ensuring that they are in compliance with the disparate terms of the various software licenses they they have agreed to. These terms may include licensed numbers of users, terms of use, etc. Failure to keep these records in an accessible and detailed form may subject the business to damages in the event of a license audit.

Provisions for license audits are usually contained in the software EULA and usually allow the vendor to audit the customer's premises for unauthorized use of the software. Such use may either force the customer to purchase additional licenses, pay damages, etc. While it is certainly possible to violate open source licenses (and be held accountable) the terms of these licenses usually only become relevant when the software is redistributed (see section 2.2).

The burden of compliance with the terms of the relevant EULA's is the main disadvantage to using proprietary software.

1.4 A Strategic Look at Proprietary Software

Because proprietary software has been dominant in most general purpose environments, most IT managers feel comfortable staying with it. However this approach does have a number of strategic implications which must be dealt with.

Often, the license agreements do not allow much flexibility regarding deployment of proprietary software. Therefore, any project requiring such software must pass through a full procurement process before it can be implemented, even for a trial program (trial copies of the software often exist, but in the event where they are untracked, they could lead to serious downtime in the event that the trial license lapses).

2 An Overview of Open Source Software

Open source software, like proprietary software, is protected by copyright law to prohibit unauthorized redistribution of the software or of derivative works. However, open source software licenses explicitly allow redistribution in certain circumstances and do not limit use via EULA's. As defined by the Open Source Initiative, Open Source Software bestows the following rights on the user:

- The right to access and modify the source code.

- The right to redistribute unmodified and modified copies of the program.

The above rights may be limited to some extent (i.e. requiring modifications to be distributed as patches, or requiring that derivative works be distributed under the same license). For the complete definition, see http://www.opensource.org/docs/definition.php.

Open source software, under this definition, can be deployed on any number of systems without violating the reserved rights of the owners (and hence risking a license audit). Furthermore, the development of the software is a community effort and fundamentally different from the development of proprietary software.

Most major open source projects are developed by a worldwide network of developers which work on the software for various reasons. Some are employed by companies which are making a strategic investment in the software, some are students wishing to improve and demonstrate their programming skills (Linux started off as a project by a Finnish university student), and others are hobbyists which contribute for the fun of it. No one single company charges licensing fees for general use and no single company bears the full cost of development. In this regard, open source software allows businesses to use the existing product free of charge and then bear the cost of developing additional features that they deem necessary. In this way, open source software passes the cost of development to the customers in ways which are fundamentally more flexible than the licensing approach allows.

2.1 Support for Open Source Software

In the absence of a single vendor for a project (often but not always the case), support can be achieved through a number of additional venues. These include email lists and newsgroups, third party consultants, and support companies. These options present different opportunities and all three are usually necessary for any reasonably large business using the software.

The support companies often exist for the purpose of offering commercial support agreements to companies which require an external emergency point of contact. These companies offer similar support services to those offered by major proprietary software manufacturers. They can often work with the original developers to develop bug fixes if necessary as well. Third party consultants are similarly able to offer the same level of support as with proprietary applications.

The major difference between open source and proprietary software is in the ability for a general user to contact the developer who contributed a specific portion of code in the event a bug is discovered and discuss possible solutions. This ability (usually available via email lists) allows support companies, and to a lesser extent, third party consultants, to offer better services than they might be able to offer if they were working with proprietary software.

2.2 Open Source Licensing Overview

Open source licenses generally fall into three groups: MIT/BSD style license, GPL-style licenses, and other licenses. These licenses place different requirements on redistribution of the source code. An understanding of these licenses are most important when redistribution of the software is contemplated, particularly where the redistribution would not be able to be released as open source software.

2.2.1 MIT and BSD-style Licenses

The MIT and BSD style licenses were originally developed by academic institutions who saw the value in sharing their work but were primarily concerned with liability. They generally allow the source code to be used as part of any program, proprietary or open source, though early versions of the BSD license contained an "obnoxious advertising clause" designed to ensure that the University of California was credited in every derived program. These licenses are considered to allow the most freedom, and could be simply summarized as "Use this source code in whatever ways you see fit, but don't sue us."

2.2.2 The Gnu GPL and LGPL

The GNU (GNU's Not UNIX, an attempt to produce a Free alternative to proprietary UNIX systems) project developed a set of licenses to compensate for perceived shortcomings in the MIT and BSD-style licenses. The founder of GNU, Richard Stallman, considered it inappropriate that proprietary software vendors could utilize the work of open source developers without any compensation. Therefore, he devised the General Public License (GPL) and Library General Public License (LGPL) which are designed to protect open source code from being used without other authorization in proprietary software.

The GPL and LGPL therefore are designed to give license for distribution of the source code if, and only if, the subsequent redistribution is licensed under the same terms.

The GPL is particularly attractive to larger companies (such as IBM) which have traditionally manufactured proprietary software, but are considering moving to open source software to reduce research and development costs. For example, the GPL prevents contributions that IBM makes to the Linux kernel from being available to Sun for their use in Solaris free of charge (though, of course, Sun could use those contributions if they wish to use Linux too, and their contributions would be available to IBM). The enforced reciprocity is therefore a benefit to those who contribute but not for those who don't.

2.2.3 Other Licenses

Many other licenses have been developed for specific purposes. These include the Qmail License, the IBM Public License, the Mozilla Public License, and many many more. These licenses are generally designed to protect the interests of the party which releases the software under while attempting to foster the sort of community which an open source project needs in order to survive and grow.

2.3 Advantages of Open Source Software

Open source software offers a major advantage regarding flexibility of deployment. With little to no requirements to track licenses (unless one is engaged in distribution of software), open source relieves a business from a few major administrative expenses.

Open source software also often offers better support when comparing other software with similar marketshare. Access to the original developers of a section of code and the code itself enables the providers of such support to ultimately do a better job.

Additionally, the financial pressure on proprietary software vendors often forces them to fix only a small percentage of bugs found. This means that customers must often have no choice but to live with problems they find. With open source software, one always has the option of hiring a developer to fix a problem or add a feature.

Finally, we believe that open source software is less expensive in the long run because the research and development is divided on an "as-needed" basis. This allows open source software to achieve greater flexibility at a lower cost in part due to the inherent competition in this system.

2.4 Disadvantages of Open Source Software

There are many markets where no adequate open source products are available.

2.5 A Strategic Look at Open Source Software

Open source software represents a fundamental shift in IT strategic thinking. Strategists which use proprietary software select vendors which they hope will continue to add meaningful value to their products in the future. In contrast, open source software represents a "development on demand" option which allows greater flexibility in developing optimal line of business applications than boxed software is.

Many line of business applications, such as CRM and ERP are traditionally heavily modified anyway. Using open source applications for the foundations of these applications may actually pay off quite well in the end.

Additionally, open source network infrastructure servers (with the possible exception of directory services) are generally more mature than their proprietary competitors. Utilizing these applications can save implementation and support costs.

3 Commercial vs. Non-commercial Open Source Software

Many companies offering support for open source software require that the software is obtained from them. This is understandable as otherwise the software could be otherwise modified and more difficult to support. The result is a form of commercial open source software. Such software is still redistributable, but the copies are not entitled to the same support as the original purchase. Examples of commercial open source software include Red Hat Enterprise Linux and Mammoth PostgreSQL. Bundles which prohibit redistribution are not considered to qualify, however.

Many companies, including Red Hat place additional EULA-like restrictions on their support services in order to ensure that the support is being used fairly. This approach does, however, essentially create an environment where the open source software is limited in how it can be implemented, and it also re-introduces the concept of license tracking, albeit in a slightly simplified way.

3.1 Advantages of Commercial Open Source Software

Commercial open source software offers a similar environment to proprietary software regarding support and restrictions regarding support. It is often the only way that service agreements can be obtained, and these agreements can be important for the most mission-critical systems. Additionally, the fact that the software is well known by the supporting entity means that support issues, when they arise, may be resolved more rapidly.

Additionally it is often relatively easy to migrate from a non-commercial package to a commercial version of the same software. This means that, where service level agreements become necessary, it is neither costly nor difficult to migrate to a supported offering.

3.2 Advantages of Non-commercial Open Source Software

Commercial open source software usually places restrictions on implementation if the support offerings are to be used. For example, Red Hat Enterprise Linux's support agreement includes an "all or nothing" clause, so if *any* Red Hat Enterprise Linux systems are to be supported at an organization, support agreements must be purchased for all of them. This does place limits on deployment which are strikingly similar to those used for proprietary software. Non-commercial open source software is free of these restrictions. Therefore, the fullest benefits of open source software (including flexibility of implementation) may only be obtained by running non-commercial open source software.

4 General Recommendations

4.1 When To Use Non-Commercial Open Source Software

Non-commercial open source software can be successfully used in a business during trial deployments, where additional flexibility is needed, or where the software is sufficiently mature as to rarely require maintenance (DNS, web, and DHCP servers, for example). Additionally, if the business has sufficient in house capability, they may opt to support their own software rather than rely on external companies for support.

Non-commercial open source software can also be used in environments where support is straightforward (Point of Sale terminals, for example).

4.2 When To Use Commercial Open Source Software

Commercial open source software should be used where the installation is stable enough to warrant the overhead of license management and where support is sufficiently necessary to justify the added expense. Datacenter environments, authentication servers, telephone switches, and other critical systems may be best run on commercial open source software, provided that they are not transient deployments, such as pilot programs.

4.3 When To Use Proprietary Software

Proprietary software should generally be used when open source options have been evaluated and determined to be inadequate.

5 Metatron Technology Consulting's Approach to Service Level Agreements

Unlike most providers of support for open source software, we will offer service level agreements even for systems running third-party non-commercial software. By doing this, we hope to allow businesses to take increasing advantage of non-commercial open source software. Ensuring that this arrangement works requires a close working relationship with our customers, as service level agreements are offered on a per system basis.

Metatron Technology Consulting's customers are entitled to outstanding support and our subscription services offer a wide range of monthly support options for businesses with or without inhouse IT departments. For more information, email us at sales@metatrontech.com.