This article was first published in ILTA's June, 2006 white paper titled "Open Source Software — The Door Is Open" *and is reprinted here with permission. For more information about ILTA, visit their website at www.iltanet.org.*

::Understanding Open Source

Like many technology trends, open source can be somewhat of a hot button issue, fanning the passions of advocates and detractors alike. Depending on the speaker, open source may be praised as an infallible panacea or condemned as an obstruction to innovation and productive free enterprise. But reality is often much more complicated than either the proponents or critics may care to admit.

Open Source Defined

Open source is software that may be distributed freely (in both source and binary form), modified and used for any purpose. The Open Source Initiative, a certification and advocacy group, argues that the accessible nature of such software encourages participatory development and results in better software, delivered more quickly. And there is evidence data to support the argument that open approaches result in winning applications and products. The Internet is powered by technologies like the Linux operating system, Apache Web server and numerous other development software and tools.

While frequently thought of as the domain of the Linux operating system, open source software can be made for any operating system — there's a substantial body of open source that runs on Windows.

The Origin of Open Source

To better understand open source, we should first understand its roots and history. Open source grew out of the free software movement founded in the mid-1980s and had its own roots in the culture and practices of academic computing in the 1960s and 1970s. In those days, at the dawn of the age of mainframes, a large number of computer scientists and programmers worked in institutions of higher learning and followed standard academic principles, sharing their tools, techniques and knowledge with their colleagues. They regularly published their code, solicited peer review and encouraged collaboration.

As commercial markets for computers matured, systems vendors discovered that revenue opportunities existed not only for selling hardware but also for licensing software, the tools and applications that could transform that hardware from a hobbyist's toy into an important business machine. A market for software created demand for skilled developers to write it. This demand resulted in lucrative job opportunities that acted as Sirens' calls to these early academic hackers. Some cut their metaphorical hair and joined the corporate world in pursuit of material riches. In the process, they often signed nondisclosure agreements with their new employers which cut them off from further collaboration with their former brothers-in-digital-arms.

But not all of them joined the dot-com revolution of its day. One MIT graduate student in particular refused to join his peers in industry. Instead, in 1984, Richard Stallman left MIT and formed the Free Software Foundation, officially beginning a mission to convince the world that proprietary software was a bad thing. Actually, he believed that closed, proprietary software was more than just a "bad thing," he believed it to be morally wrong and unethical. Indeed, in many respects the mission of the Free Software Foundation bears comparison to a crusade, complete with religious fervor and undertones.[1]

Not everyone who created and shared software agreed with Stallman's philosophy. By the late 1990s, free software had grown increasingly popular and was in use not only by academics and hobbyists but increasingly in corporate and commercial environments. Many associated with such software, including businesses that created, sold or supported it, felt the rhetoric and reputation associated with "free software" was impeding its broader adoption. Some people, they felt, were put off by the prospect of being associated with this arguably radical movement. So they removed the rhetoric and rebranded "free software" as "open source," taking the moral and ethical issues off the table and advocated the practical benefits of open development.

Understanding the Basics of OSS Licenses

There's more to the history of the free software and open source movements[2], but for practical purposes, understanding open source really means understanding open source licenses. Wikipedia.org defines a software license as:

> "...a type of proprietary or gratuitous license [a document, contract or agreement giving permission for an individual or entity to do something] as well as a memorandum of contract between a producer and a user of computer software — sometimes called an End User License Agreement (EULA) — that specifies the perimeters of the permission granted by the owner to the user."[3]

There are a number of open source licenses, but they basically fall into two general categories: Copyleft and non-Copyleft. Copyleft is a concept created by Richard Stallman (it's designed to be a pun on the word "copyright"). The best example of a Copyleft license is the GPL (GNU General Public License). In lay terms, Copyleft says that if you make changes to an open source program and distribute the resultant derivative work, you have to make the source code of that new work available under the same license so that others could similarly share and modify the software.[4]

Can Open Source Licenses Be "Closed"?

Importantly, software that is published under a non-Copyleft open source license has no such source redistribution requirements. This is an important and subtle fact that is often overlooked by those arguing for or against open source. Thus, it's actually possible to have open source software that becomes incorporated into proprietary software, subject to all the restrictions one might find in a vendor's standard license agreement.

When conducting due diligence as part of an RFP or evaluation process, you should consider open source using the same rigorous standards you'd apply to any vendor or technology.

One company that has used open source software in this way is Microsoft. In 2001, the *Wall Street Journal* reported that the company had ". . . been quietly using such free computer code in several major products . . ."[5] As it turns out, Microsoft incorporated networking code from the FreeBSD UNIX operating system in Windows 2000 and other products. More recently, Microsoft has adopted similarly non-Copyleft-licensed IP in the form of message-passing technology called MPICH2 in its Windows Server 2003 line of products[6] and through its use of the Kerberos authentication software specification.

The Effects of OSS Licensing Issues

In the early days of open source, businesses were often concerned that using open source would require them to share their intellectual property with outsiders and competitors. But in practice, licensing issues don't affect firms using open source software very much at all.[7] Consider this scenario: Your firm brings an open source application in-house and modifies it heavily — say it extends and incorporates it into a homegrown application such as a client extranet, an application many firms consider a competitive differentiator. Does your firm have to share its new creation with the world?

No, it doesn't. The reason is that in open source licenses with sharing clauses, those requirements only kick in upon distribution. If you were to package and distribute a binary file of your new creation to others, it might be a different story. But since you are not distributing software (hosted access doesn't qualify as distribution), your firm is not subject to the sharing requirements.

(Of course, those in the open source and free software communities would argue that sharing is good and will result in better software for everyone. But if a firm makes internal changes it feels grant it a competitive advantage, it's perfectly able to keep those to itself.)

Understanding Open Source Myths and FUD

The term "FUD" (fear, uncertainty and doubt) is likely familiar to anyone who's ever read a technology trade publication. Coined in the 1970s in response to perceived aggressive sales tactics by IBM, FUD is technology spin. This article has already dispelled a few common myths, and on closer examination, much of the FUD associated with open source doesn't hold water. Common anti-open source spin includes:

It's anti-intellectual property. Actually, open source depends on intellectual property rights like copyright, which allows authors to grant others the right to use works only in accordance with specific terms.

It forces organizations to share their IP. Only some licenses require sharing of modifications and changes and only when the resultant work is distributed to others.

It's lower quality than proprietary software. Software can be good or bad and should be judged independently of the license under which it's made available. Indeed, many advocates of proprietary technologies debate each other similarly. Just as with its proprietary counterparts, there is stable open source software as well as unstable.

It's unsupported. Support options for open source vary. There are a number of commercial products supported by vendors such as IBM and HP. There are also projects that are supported by communities of developers and end users. Sometimes, community support can deliver more expertise, in a faster timeframe than outsourced vendor support for proprietary products. But as with any software generalization, these examples can cut both ways.

There's no application that does "X." Open source has excelled and received the most attention for its success as infrastructure technology. But an examination of its growth during the last five years shows numerous emerging, commercial, vendor-supported offerings for CRM, BI, ERP, database, development, productivity and other areas. The real question is "does the lack of an open source application for X mean one shouldn't look at candidates for Y?"

It's not really free. This is completely true. Nothing is really free. When evaluating the total cost of ownership (TCO), organizations need to consider license fees, support and maintenance fees, installation, training, customization, consulting and other external/internal costs.

The Hidden Open Source in Legal

If your firm is like most, it's probably not running the open source OpenOffice.org productivity suite on Linux desktop workstations. But chances are your firm is already using open source, probably without realizing it. Perhaps more aptly-named "hidden source," this code inhabits a number of applications and is supported by several well-known vendors.

Companies such as ADERANT and Cisco include or support applications such as Tomcat, which provides various Java technology support for Web server platforms including Windows Internet Information Server (IIS). Even more deeply hidden, if your organization is using Network Attached Storage (NAS), or a spam/firewall solution from a company such as Barracuda Networks, it's likely using open source as well. (Of course, these later products are delivered as appliances, and the vendor supports everything under the hood, making them easier to manage, maintain and upgrade than traditional software.)

Where Are the Open Source Applications in Legal?

While open source has flourished in some areas, it's safe to say that it's primarily serving a niche role in legal environments. Key applications in the legal IT ecosystem include:

Asterisk — A software-based PBX system offering support for VoIP and standard telephony equipment

Acrophobia — A network PDF printer created by ILTA's Open Source Peer Group

Snort — An intrusion detection tool

Several content management projects suited for intranet / extranet applications

Anecdotally, one tends to find open source used more by smaller firms. Several factors may explain this. For one, these organizations may have smaller proportional IT budgets compared to larger firms. A greater share of these resources may also be dedicated to maintaining investments in proprietary, commercial applications. These constraints may create incentives to explore open source as a cost control measure or as the only path to implementing certain projects.

Also, in smaller and mid-sized organizations, overall IT staffing and staff-to-timekeeper ratios are lower.[8] While this limits overall IT capacity, it also means staff may be called on to cover more functional breadth and may have greater interest in exploring new solutions that are more accessible in terms of cost or customization.

Finally, in smaller environments, change management processes are often less involved, meaning an enterprising technologist can more easily secure permission to deploy a new solution or even "fly under the radar" until a working proof of concept is in place.

Within ILTA, there is a growing open source community and the Open Source Peer Group. According to its leadership, the group consists of over 150 members, primarily at small and mid-sized firms. This group seeks to explore, discover, promote and even create open source technology that fits the needs of the legal community.

Is Open Source for My Firm?

Even if your organization is not looking to blaze new technology trails, there are several reasons to include open source candidates as potential solutions or components of IT projects. Of course, when conducting due diligence as part of an RFP or evaluation process, you should consider open source using the same rigorous standards you'd apply to any vendor or technology. If it doesn't pass, your organization should take a pass. Your evaluation requirements will likely vary based on the project at hand and organizational priorities, but some to consider are:

How well does a particular application suit our needs? Does it meet functional requirements? What are its strengths and weaknesses compared to other candidates?

Are we comfortable with our support options? If having a business to contact directly is important, make it a requirement. Several open source technologies are delivered or supported by stable, for-profit companies. If you're comfortable with support delivered through a developer and user community, make sure a vibrant one exists. (Thankfully, their mailing lists are usually open, so you can investigate directly.)

Is this technology compatible with the firm environment and IT and user staff skills? Will the technology fit well within your existing environment?

How does the TCO stack up? Beyond functionality, how do the ease of implementation, ease of management, support and maintenance requirements for this software compare to other options?

Open source advocates would encourage firms to include a comparison of some of the built-in benefits of open source that proprietary providers can be hard pressed to match including:

How easy is this application to customize? Having direct access to the source code and unhindered access to the core development team of the project can provide organizations with greater freedom to modify, customize and extend their software as needed.

How quickly are patches and bug fixes addressed? A broader community with source access will often diagnose, repair and distribute big fixes with greater speed and efficiency than a single entity.

What assurances for long-term support and stability can the vendor provide? If the company changes product focus, tries to end-of-life a product its customers are not quite ready to upgrade, or goes out of business, organizations may be left with legacy systems no one can maintain, modify or support.

The Secret Benefit of Open Source

There's secret value to open source available even to firms that are dubious about using free software, and that's vendor leverage. For example, it can be hard for vendors to compete with "free." Consider a common scenario: Your organization is evaluating a new piece of technology. It has identified an open source equivalent that is reasonably comparable to proprietary offerings or close enough to warrant investigation.

Add to this mix two vendors who are competing for your business. Each promises maximum functionality, future development and top-grade support. You have an advantage in that interaction because you have choice. Potential vendors will be forced to compete on price, service or other benefits. It will make them work hard to earn your business and to keep it.

Now imagine you sit both representatives in a room and tell them that there is free software that will meet some reasonable percentage (or all) of the functional requirements for your project. The software doesn't have a commercial support entity, so your TCO will include internal staff training and a bit more maintenance effort. But it's looking like your TCO might be interestingly lower than either of their packages. Then, you send each on their way with the advice that they reconsider their proposals and see what they can do to make their offers more competitive.

You may be able to benefit from open source without actually using it.

The Real Topic for Concern — Open Standards

Your organization may be an active proponent of open source, may be cautiously exploring it, or may still be dubious of its value and applicability within your environment. And that's okay — at its heart, open source is about choice. In fact, this choice is the fundamental ideological difference between open source and the Free Software Movement that spawned it — the ability to make a practical decision based on what best suits the needs of your organization — even the choice not to use open source.

But there is one issue tied closely to open source that all organizations should choose to pay close attention to, open standards. This issue is often linked to open source because the very nature of open source tends to ensure standards compliance — if an open source technology isn't compliant and is in wide use, there are those who will modify it and make it so.

Those who witnessed the early days of electronic mail remember what life was like before internet e-mail standards were widespread — proprietary e-mail providers with closed networks made it difficult, if not impossible, for users to communicate outside their networks. It was open standards (and open source technologies implementing those standards) that created a bridge to connect people and information and transformed e-mail into the pervasive tool it is today.

Today, applications that use open standards are easier to integrate with each other, and the existence of several key open projects puts pressure on vendors to respect and support standards. In legal IT, standards support is vitally important, especially as new technologies such as Web services gain traction. Firms commonly expend tremendous resources to ensure their applications can co-exist and interoperate effectively. Standards-compliant technologies can go a long way toward controlling those resource expenditures.

But that's a topic for another white paper…

Endnotes

1 For an amusing portrait of Stallman highlighting these religious undertones, see: www.tinyurl.com/n8wd4.

2 See the excellent documentary "Revolution OS," available via Netflix.com or Blockbuster.com.

3 See: www.wikipedia.org/wiki/Software_license.

4 A good example of this principle in practice is my TiVo video recorder. TiVo uses a version of the GPL-licensed Linux operating system that it has modified. Because it's distributing that Copyleft software as part of its product, it also must make its modifications available, which it does at: www.tivo.com/linux.

5 See ZDnet: www.tinyurl.com/hty4t.

6 See eWeek: www.tinyurl.com/ley2a.

7 Firms concerned about the license of software they wish to use in-house should review that license with the same diligence as they would any vendor's end user license agreement.

8 See *ILTA's 2005 Technology Survey*.