OSBR.CA

The Open Source Business Resource

SEPTEMBER 2007

SEPTEMBER 2007

PUBLISHER:
The Open Source Business Resource is a monthly publication of the Talent First Network. Archives are available at the website: http://www.osbr.ca

EDITOR:
Dru Lavigne
dru@osbr.ca

ISSN:
1913-6102

ADVERTISING:
Rowland Few
rowland@osbr.ca

GRAPHICS:
Ryan May

In his book Foresight and Understanding: An Inquiry into the Aims of Science (ISBN 0-313-23345-4), Stephen Toulmin wrote "Definitions are like belts. The shorter they are, the more elastic they need to be. A short belt reveals nothing about its wearer: by stretching, it can be made to fit almost anybody."

Keep in mind the nature of elasticity while reading through this issue of the OSBR. The theme this month is "Defining Open Source"; however, you'll find that the articles build upon and extend both the Open Source Definition or OSD (http://www.opensource.org/docs/osd) and the Free Software Definition (http://www.fsf.org/licensing/essays/free-sw.html). This stretching in order to fit almost anybody is bound to make the open source purist uncomfortable; it is our intent to provoke thought and we look forward to receiving and publishing reader feedback.

Russ Nelson of the Open Source Initiative provides a historical perspective on the creation of the OSD and some hints towards its future direction. Tony Bailetti and Peter Hoddinott take a more radical yet practical approach on the meaning of "open" and "source" and how "open source" differs from "open code". Patrick McNamara of the Open Hardware Foundation provides a compelling comparison of how "open source" as it applies to software can also be applied to hardware. Monica Mora provides insight into the Open Education Resources movement which is applying open source software methodology to the collaborative creation and distribution of knowledge content. Finally, no discussion on open source can be considered complete without some insight into the communities which create open source assets. Nelson Ko provides tips for business to benefit from interacting with open source communities while avoiding culture clash.

As a business person you may be thinking "why should I care about the semantics of a definition or what the term "open source" is being applied to? Let the open source people fight it out amongst themselves while I concentrate on the business of making money." That would be good advice if the following was not true: what is commonly thought of as open source left the realm of the philosopher and the hobbyist programmer several years ago.

Further, movements which are largely still being defined are gaining momentum and are changing the rules of economics for government, academia, and the enterprise. The astute within the open source and business communities already recognize this and are positioning themselves to benefit from the new definitions. The better question to ask yourself is "once the new paradigm has been established, will my organization emerge as passive observer or as active participant?"

Dru Lavigne,

Editor-in-Chief

Dru Lavigne is a technical writer and IT consultant who has been active with open source communities since the mid-1990s. She writes regularly for O'Reilly and DNSStuff.com and is author of the books BSD Hacks and The Best of FreeBSD Basics.

"*At OSI we have seen that the process of growth in Open Source is more evolutionary than revolutionary. We invite public debate with each successive wave of newcomers to start the process to close the gap between what they imagine Open Source to be and the reality of what is required (and why).*"

Danese Cooper, Board Member of the Open Source Initiative

The Open Source Initiative or OSI (http://www.opensource.org) and the Free Software Foundation or FSF (http://www.fsf.org) share a common goal: that everyone should be free to modify and redistribute the software they commonly use. 'Should' is of course a normative word. For the FSF, 'should' is a moral imperative. Anything else is an immoral restriction on people's activities, just as are restrictions on speech, press, movement, and religion. For the OSI, freedom is a necessary precondition for a world where "software doesn't suck", in the words of a founder of the OSI.

The FSF started from its founder's GNU Manifesto widely published in 1985 (http://www.gnu.org/gnu/manifesto.html). Given the manifesto's hostility to copyright, and given the failure of the Free Software Foundation to gain any traction amongst commercial users of software even with a 13-year head start, a group of people gathered together in 1998 to talk about a new strategy to get the corporate world to listen to hackers. They were impressed by Eric Raymond's Cathedral and the Bazaar's take-up among business leaders (http://catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/).

Origins of the Open Source Definition

The term 'open source' emerged from the efforts of this group of people frustrated by the multiple meanings of 'free' and its failure as a marketing term.

On the one hand, 'free' is good, because everybody wants to get something for free. To the Free Software Foundation 'free' does not mean the absence of a price tag; they want 'free' (http://www.gnu.org/philosophy/free-sw.html) to mean freedom. Further, when businessmen look at what they get for 'free', they may think 'worth what I spent' and equate zero cost with zero value. The term 'open source' aims to lose that luggage.

The other source of frustration has been the insistence that the free software movement is about morality and that 'free' software is the only moral choice. As soon as you say that, you put off people who disagree. "Oh, you mean I'm a bad person?" "No, you're just doing a bad thing." "As if that's any better!" If you want to sell a foreign idea to people, you have to start by not judging them as immoral.

Two members of the group that picked the name open source as a marketing term for free software were more excited than the others: Eric Raymond, author of the Cathedral and the Bazaar, and Bruce Perens, former leader of the Debian GNU/Linux project. They wanted to be an organization, not just two individuals, so they sought board members to guide an organization to stand behind their efforts. As a marketing term, 'open source' is great; they decided to harden its meaning a bit, by seeking a trademark to be licensed to anyone who met a fixed definition. Bruce had put a lot of effort into Debian's Free Software Guidelines or DFSG (http://www.debian.org/social_contract.html#guidelines), so he imported those guidelines into the Open Source Definition or OSD (http://www.opensource.org/docs/osd).

In time, the decisions to seek a trademark and be based on the DFSG would be seen as flawed. The United States Patent and Trademark Office objected to 'open source' as being too descriptive, and refused to grant the OSI a trademark. As the OSI discovered, a trademark cannot simply describe the good or service being trademarked; there needs to be more creativity involved. While open source is very descriptive, which is part of its power and attractiveness, a good trademark it is not. Instead, the OSI established the "OSI Open Source Approved License" standard.

Reuse of the DFSG also proved problematic. First, it created resentment on the part of Debian, whose members felt it was improperly reused. Second, it was never meant to be a bulwark against hostile redefiners. Many would-be open source abusers have pointed out that the OSD doesn't require distribution of the source code. In the form of guidelines for Debian, the DFSG never needed such completeness. Due to its culture and underlying philosophy, the Debian project would never begin to consider the inclusion of software without source code.

Definition Challenges

Since its inception, the OSD has needed some terms added to it, and some terms revised. For example, when some open sourcers threatened to require that redistributors acquire positive acceptance of the license by click-through, we added OSD #10, which prohibits license stewards from requiring any particular technology in licensing. The difficulty from the licensor's point of view is that court precedents existed to defend click-through licensing. The difficulty from the licensee's point of view is the impossibility of clicking through every license in a distribution.

Some would-be software distributors thought that they could simply not include the source, and still use the open source label. We clarified our intent in OSD #2 which states that "The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed."

OSD #9 originally stated that the "License Must Not Contaminate Other Software". However, what does "Contaminate" mean? It sounded like imprecise language, so we changed it to say "Restrict" instead.

Even Debian has seen the need to change their DFSG. Their community process is such that any change would be painful. So, rather than change the text of the DFSG, the people on the debian-legal mailing list have added extra requirements which exist as a form of case law. If you try to submit software that only complies with the DFSG, it still might not get into Debian because of these extra requirements.

There was a private effort made a few years ago between Software In The Public Interest (the legal owner of Debian) and the OSI to try to reconcile the DFSG with the OSD. This failed because of the conflicting goals of the two documents. Although textually similar, the DFSG is used to decide what software goes into Debian, while the OSD is used to decide what software gets to claim to be open source software.

One thing that readers must know is that while the OSI tries to work with license stewards to get the best possible license, we don't require it. We've approved licenses where we've all looked at each other, shook our heads sadly and said "That license just isn't gonna be widely used." But we're not infinitely wise, so we gave the license its due as abiding by the definition of open source.

Sometimes, license stewards approach the OSI with flexibility in mind. I recall a conference call with the lawyers at NASA. They worked with us to ensure that the NASA Open Source License would work not just for NASA, but for any U.S. Federal Government agency wanting to release software as open source.

Microsoft, to their credit, produced a fairly open-source-sounding set of licenses in their Ms-Pl (http://www.microsoft.com/resources/sharedsource/licensingbasics/permissivelicense.mspx) and Ms-Cl (http://www.microsoft.com/resources/sharedsource/licensingbasics/communitylicense.mspx) without needing to consult with us. They did ask about any speedbumps when they submitted their licenses for approval. Of course they know full well their reputation in the open source community, so they didn't want to make any trivially avoidable mistakes in their submission.

Beyond the Definition

Open source has proven to be such an alluring concept that you see open source hardware, open source radio, and open source movies. The concept is applicable to anything which has separable design elements. So open source hardware would be fully documented, modular, with separate parts available for sale. The Lego Group has adopted this idea with its Lego Mindstorms and NXT products. People have created alternate devices and even alternate firmware to download into this hardware.

Open source radio acknowledges that radio, while a stream of audio, occurs in a context, and the quality is improved when that context is open to contributions from all listeners. Open source movies such as Elephants Dream (http://www.elephantsdream.org/), consist of many elements such as audio tracks, animation models, textures, and plot. By distributing the movie not just as a finished product, but in separable design elements, watchers can contribute changes, or even compose their own movie from the elements.

Our success in promoting open source processes is a blessing and a curse. The more people who understand what open source means, the less likely anyone is to succeed at corrupting the term. On the other hand, we risk losing control over the use of the term and thus the meaning of it. That's why we have the "OSI Approved Open Source" program.

We've been so successful in propagating the open source brand that some people say there are too many open source licenses. From the perspective of somebody who wants to redistribute a distribution, this makes sense. They have to do the due diligence of researching the import of every license on every piece of software that the distribution includes.

We appreciate both views: that there aren't enough open source licenses and that there are too many licenses. Even Microsoft wants their own open source license pair (public and community) approved. We try to seek a balance by assigning attributes to licenses depending on what categories they fit into. We've even convinced some license stewards to deprecate the use of their licenses; they're still open source licenses, but they shouldn't be used.

All this open sourcing is also at risk from patents on ideas rather than mechanisms. Software being fundamentally an idea, it seems improper to patent it. Yet that's where we are, and the patent juggernaut seems to roll on. The patent system is supposed to be used to protect mechanisms, so that people who invest in building these mechanisms can have exclusive rights to these mechanisms.

Unfortunately in the USA, a mechanism which can be built using software has been interpreted to mean that software can be patented. Unfortunately, many previously invented software mechanisms have been patented. Software patents are allowed even when they are obvious to any skilled practitioner of the art. This is very bad when software patents get written into standards documents, because that prohibits open source implementations. We're pushing back with our "Open Standards Requirement for Software" (http://www.opensource.org/osr). This is a new program whose details are not completely settled yet.

You can help us. Look for the OSI Open Source Approved License on your software. We're a non-profit charity, so you can contribute, which helps our advocacy efforts. And when the Open Standards Requirement is firmed up, you can ask for the Open Standards Requirement for Software when you adopt standards.

Recommended Reading

Perens, Bruce, The Open Source Definition (http://www.oreilly.com/catalog/opensources/book/perens.html)

Stallman, Richard M., Why "Free Software" is better than "Open Source" (http://www.fsf.org/licensing/essays/free-software-for-freedom.html)

Russell Nelson is a founding board member of the Open Source Initiative. Although a board member, he's just one board member and the opinions expressed herein are his own, and not official board positions. He has been writing open source before we started calling it open source. His software has made it into McDonald's cash registers, operating rooms, and aircraft flight control systems. At one point, his GPL'ed packet drivers were arguably running on more CPUs than anything the Free Software Foundation had written. This isn't the case anymore, of course.

"*Most people consider a company OSS when it contributes code to an OSS project, but nowadays a significant value of open source lies in non-code contributions...We should start thinking more about how to study non-code contributions, and how this relates to the commercialization of open source projects (and not only software).*"

Carlo Daffara, the Italian representative of the European Working Group on Libre software

The Open Source Definition or OSD (http://www.opensource.org/docs/osd) defines the criteria to which the software must comply for it to be deemed to be open source software. The term "open source", however, is used to label a broad assortment of phenomena that fall well outside the established OSD. In addition, there is ambiguity in what is meant to be covered by the terms "source" and "open".

We envisage a definition of open source that equally applies to software, hardware schematics, content, and processes, not just software.

To advance our understanding of open source, we argue that we need to:

- Define open source in terms of the attributes of the systems used to produce, use, and distribute assets
- Act to strengthen the soundness, vitality, and proper functioning of these systems

What is Meant by "Source"?

The term "open source" was invented as a marketing term in 1998. Proponents of the term "open source" successfully argued that the term "free software" was fraught with challenges that included it being ambiguous and it being disliked by corporations.

Several months ago, a talented software professional challenged our use of the term open source when referring to a software application released under the BSD licence, an open source licence which complies with the OSD. The professional argued that the software was "open code", but not "open source". Intrigued, we asked for clarification.

It was explained to us that the code was produced by a single private organization that periodically bundled together a release and published it on Sourceforge.net. While the code was released under the BSD license, the production of the code lacked key open source characteristics. Instead, it provided:

- No external contributors – all code was developed in-house prior to being published on the Internet
- No visibility of who developed the code and when they developed it
- No mechanisms to the general public for (i) contributing to the production of the code prior to its release in Sourceforge.net or (ii) participating in the governance structure of the orgnization that produced it

In short, the code was open, but the process used to produce it was closed. There was no public community behind the production of the code, and no accommodation for such a community.

Our discussion then turned to the ambiguity in the usage of the word "source". Does source mean the computer code written in a recognized programming language, or the process used to produce the code, or something else?

If we allow “source” to mean two different things: (i) the process used to produce the code, and (ii) the computer code, four cases are possible:

1. Open process and open computer code

2. Closed process and open computer code

3. Open process and closed computer code

4. Closed process and closed computer code

We surmise that many use the term “open source” with case 1 in mind. For example, the Eclipse code is licensed under the OSI approved Eclipse Public License (EPL). The central repository for the code base is available and the general public can, with ease, track the pedigree of the code. Release dates are known and published. The general public is encouraged to contribute to the organization itself, to define projects, and to write code. Moreover, the governance structure used to manage all the Eclipse projects is transparent.

Case 2 was outlined above, and it characterizes what our software professional called “open code” but not "open source". A fundamental contradiction seems to exist when an open source asset is developed using a process controlled by a single party. For example, the Open Office project has been criticized for encouraging a development culture that differs radically from the open-source norm (http://www.computerworld.com/action/article.do?command=printArticleBasic&articleId=9037499). The majority of the contributors to the Open Office project work for Sun Microsystems.

Case 3 includes instances of organizations and individuals that produce a reference implementation of some standard to accelerate that standard’s adoption. The code of the reference implementation is typically produced using open processes but the code itself may not be released under an open source licence. The reason is straightforward: releasing the code under an open source licence would potentially weaken the purpose and authority of the standard by facilitating the ease by which deviations of the standard may be introduced.

Case 4 includes instances which are typically referred to as proprietary or closed software. The process used to produce the code is closed, and the software is released using a non-open source licence. This case includes instances where several organizations create a consortium to produce code that only those with membership within the consortium have visibility and access to. Typically, such consortiums have a tiered membership that stipulates the members’ rights with respect to the code.

It can also be argued that the use of “source” does not distinguish between the three types of “code”, any of which could be open or closed. Source can mean: (i) the code used to implement a system or component, (ii) the interface where what we open is the application programming interface (API), or (iii) the data underlying the implementation where most any application or system creates value from the underlying data.

It can also be argued that source is not limited to computer code. One could extend the four cases described above to combine open and closed processes with hardware schematics or documentation, two examples of assets which are increasingly being considered as open.

What is Meant by "Open"?

More recently, we had occasion to find ourselves struggling as we tried to make sense of instances of the word "open" in the context of community code. For example, we found instances where "open" meant that releases of the code were made available to the general public (i.e., non-members of a consortium); however, releases to the general public were delayed 12 months from the time it was available to the members of the consortium. We also found instances where what "open" meant depended upon the level of membership. The more expensive memberships provided these members more privileges to participate in and influence the processes, for example with veto power. In these examples, open is not equated with full access; instead, open is a matter of degree and that degree is metered out in a distinctly defined hierarchy of privilege.

This seeming confusion and differences about what is "open" and what is "source" and the use of "open source" to refer to phenomena that fall well outside the OSD, led us to conclude that we need to better understand the characteristics of the systems in which open source assets are produced, used and distributed.

We conceptualize any such system as being comprised of four components:

1. Network: the network of individuals and organizations that produce, use and distribute an asset

2. Processes: the processes, approaches, rules and understandings that lead to the production, use and distribution of an asset

3. Governance: the governance structure of the organization and the projects within the organization

4. Value: value created through collaboration and value appropriated through competition

Metrics of Health

We observe that a healthy open source system is required to compete with a strong proprietary system; that is, an open source system cannot compete by virtue of the distribution license alone.

What one would generally agree upon as being an open source system could be expressed in terms of the health of its four components. Such a definition would not be static, but would arguably be more accurate and useful in determining the true value of an open source system. For example, one could speak in terms of a system not having reached the status of being open source until it is deemed to be "healthy". And an open source system may subsequently cease to be an open source system if its health deteriorates beyond some point. This line of inquiry could then further leverage "health" of the four components as a means for distinguishing other types of systems such as closed systems and community systems.

Our initial suggestion as to what is relevant for assessing the health of each of the four components of a system is as follows:

1. Network: large, distributed and diverse. We distinguish between an asset produced by a well developed network from an asset produced by a small number of collocated producers who have similar characteristics. A general reference model for an open source asset would be one that is produced by a well developed network that is able to integrate, test, and quality-assure contributions from a large number of diverse individuals and organizations dispersed throughout the world

2. Process: includes meritocracy where one is recognized for the quality of contributions; transparency in communications and guidelines; recruitment and promotion methods; and mechanisms for dealing with difficult people

3. Governance: includes participation; relationship between contribution and the influence that can be asserted; membership's influence over a project, influence over the overall system governance, and ability to alter the governance structure

4. Value creation and appropriation: usefulness of the asset; how free-riders are addressed--if it is too easy to appropriate value no one would pay for a membership or undergo an apprenticeship to move from being a developer/contributor who writes code or documentation to a committer with write access to the codebase; access to the asset by virtue of the license

Various other metrics can also be used. What is needed is agreement on the key ones.

Conclusion

While the OSD is useful in promoting a brand and defining the rules licenses must adhere to in order to be considered open source, there is much value to conceptualize open source as part of a larger system which describes the production, distribution, and use of an asset. We envisage four components of the system: (i) network, (ii) process, (iii) governance, and (iv) value created and appropriated. Moreover, we suggest that an asset becomes an open source asset when it is produced, used and distributed within a system that is healthy in terms of these four dimensions.

We also envisage each component to be multidimensional and identify some of the dimensions that could be used to track system healthiness.

Finally, we argue for a definition of open source which is independent of the basic structure of the asset. We envisage a definition of open source that equally applies to software, hardware schematics, content, and processes.

We invite the readership of the OSBR to embark upon a discussion of the proposed positioning of the open source system and the components and metrics identified.

Tony Bailetti holds a faculty appointment in both the Department of Systems and Computer Engineering and the Eric Sprott School of Business at Carleton University, Ottawa, Canada. Professor Bailetti is the Director of the Talent First Network. Until September 2007, he was the Director of the Technology Innovation Management program. He has taught for the Executive M.B.A. program offered by Queen's University in Ottawa since 1996.

Peter Hoddinott has over 25 years of experience in the Information and Communications industry. Peter has a B.Sc. and a M.Sc. in Computer Science, and recently completed the Technology Innovation Management program at Carleton. He is currently employed by Carleton where he works full time on advancing the objectives of the Talent First Network.

"Open Hardware is a thing - a physical artifact, either electrical or mechanical - whose design information is available to, and usable by, the public in a way that allows anyone to make, modify, distribute, and use that thing."

The TAPR Open Hardware License (http://www.tapr.org/ohl.html)

"Open source hardware refers to computer and electronic hardware that is designed in the same fashion as free and open source software. Open source hardware is part of the open source culture that takes the open source ideas to fields other than software."

Wikipedia (http://en.wikipedia.org/wiki/Open_source_hardware)

I have been involved in a number of debates on what exactly constitutes open hardware. While the definition is a bit harder to pin down than that of open source software, I believe hardware can be loosely placed into four categories of openness. They are, in order of least to most open: Closed, Open Interface, Open Design, and Open Implementation.

Closed: closed hardware is any hardware for which the creator of the hardware will not release information on how to make normal use of the hardware, in such a way that that information may be freely shared with others. A sure sign of closed hardware is requiring the signing of an NDA to receive documentation on how to make use of a device.

"Whether or not a hardware device's internal design is free, it is absolutely vital for its interface specifications to be free. We can't write free software to run the hardware without knowing how to operate it. (Selling a piece of hardware, and refusing to tell the customer how to use it, strikes me as unconscionable.) But that is another issue."

Richard M Stallman

Open Interface: in the case of open interface hardware, all the documentation on how to make a piece of hardware perform the function for which it is designed is available. In the case of computer hardware, this means that all the information necessary to produce fully functional drivers is available. This is the minimum level of openness that makes hardware useful to the open software community. Surprisingly, large amounts of integrated circuits fall into this category. Any device for which you can get a complete data sheet from the manufacturer, with no limitations on sharing the data contained within, meets the open interface definition.

Open Design: open design hardware is hardware in which enough detailed documentation is provided that a functionally compatible device could be created by a third party. It is not at all uncommon for the programmer's guides for a microcontroller to have complete instruction encoding formats, memory maps, block diagrams of the processor core, and other technical details that would make it possible to reproduce a compatible microcontroller. Open design hardware allows you to see what was implemented and what it should do, but still keeps the finer details of how it was implemented closed.

Open Implementation: hardware for which the complete bill of materials necessary to construct the device is available fall into the category of open implementation. In the realm of computer chips, this means the hardware definition language description of the device is available. For a circuit board, this would include the schematic. Everything needed to reproduce an exact copy of a device is available. This is the hardware parallel to the concept of open source software.

The debate between 'open' and 'free' (libre) that exists in the software space exists for hardware as well. In this regard, the only hardware that can truly be claimed to be free, in the same manner that the Free Software Foundation defines free, is that which falls into the Open Implementation category.

Unfortunately, unlike software, an idea and the desire to produce a hardware device that is free and open is not sufficient. Certainly in the semiconductor space, the ability to do so is beyond the individual and in most cases, beyond even a reasonably equipped development group.

Why Open Hardware?

Lourens Veen, a member of the board of directors for the OHF or Open Hardware Foundation (http://www.openhardwarefoundation.org) summarized the answer to this question as follows: “Essentially, this is a problem of freedom. We users want to be free to use the objects we own for any purpose and in combination with any other objects or software we choose or create. We should not be limited to using it only in ways that the manufacturer or some other, external, entity deems appropriate.”

Several benefits can be achieved when hardware is made open to its users:

You can use it as you see fit: the quote from Lourens is actually a paraphrase of freedom 0 from the Free Software Foundation (FSF): the freedom to use software however you see fit, made in the context of hardware. This is effectively the philosophical underpinning for “Why Open Hardware?” The remainder of the Free Software Foundation's freedoms apply, to some degree, to the hardware realm as well.

You can figure out how it really works: this corresponds to the FSF freedom 1: the freedom to study how something works and to adapt it to your own needs. Far from being a simple matter of curiosity, being able to understand how a device works can enable you to make much better use of it. For instance, if there are two ways of performing the same operation in a device, being able to understand the internal operation allows you to determine the more efficient of the two ways for a given situation.

You can make it better for everyone: this echoes the FSF freedom 3: the freedom to improve the hardware and to release your improvements so that others may benefit as well. This is the most altruistic of the philosophical reasons for open hardware. Much of the innovation throughout history has been due to individuals building on preexisting ideas and sharing the results. Building on preexisting hardware is no different.

You will notice I skipped over freedom 2: the freedom to redistribute copies so you can help your neighbor. It doesn't map quite so well into the hardware world; not for philosophical reasons so much as for practical ones. In the hardware world, especially that of semiconductors, the financial barrier to making copies of hardware is such that redistributing physical copies is not generally viable for an individual. In order to enable freedom 2, you have to embrace a number of more practical, and commercially interesting reasons for using open hardware. These are the reasons a business might be interested in producing a device based on open hardware.

Open hardware can sell more units: by making a device open, you gain access to market segments that would not be available otherwise.

Over the past few years a number of home firewall/router devices have been found to be running a version of Linux. Further, in some cases, it is possible to modify the operating system running on these devices to allow them to provide other functionality potentially unrelated to their original purpose. In these cases, the ability to modify the functionality of the device has been discovered by reverse engineering. Still, you now have people buying units, sometimes several, to use for other reasons. A unit with a lower barrier to modification, due to available open documentation, will generate an even higher level of interest within certain groups leading to sales that otherwise would not have occurred.

Open hardware has the potential to speed development of new devices: most complex hardware devices are made up of many smaller building blocks many of which are not specific to that device, just like most programs use general purpose libraries for many functions. Large hardware companies build up libraries of hardware building blocks over time, but in many cases multiple companies end up re-implementing the same basic hardware blocks. As an analogy, I don't write my own SSL library when I need SSL functionality in software; I go pull down the OpenSSL library and use it. Why should I re-design a hardware multiplier when I need one for a micro controller ALU (Arithmetic Logic Unit)? It should be noted that this effect of open hardware, reducing design and implementation time by providing readily-available libraries, tends to benefit small companies more than large ones. This can have the effect of reducing the barrier to entry into a market segment and allowing more resources to be focused on the innovative part of the product which in turn helps increase competition.

The community will help you support your product: when talking about personal computer hardware, you run into the problem of drivers. Anyone who has tried to use cutting edge hardware in Linux, BSD, OpenSolaris, Plan 9, or other more esoteric operating systems is keenly aware that much new hardware is supported poorly, or not at all. Many companies may not see a sufficient ROI (Return On Investment) for developing drivers in house for non-Windows operating systems. And in truth, it may not be financially justifiable.

However, if I as an end user cannot use brand X hardware on my nice shiny new Linux box because I can't get drivers, I am going to go buy brand Y hardware for which I can get good, working drivers even if the brand Y hardware provides less functionality. The company making brand X cards just lost a sale, all because there weren't drivers available. This is not due to a lack of people willing to write drivers for such hardware. It is due to a lack of the necessary documentation on how to make the hardware work.

Why the Open Hardware Foundation?

The Open Graphics Project or OGP (http://wiki.opengraphics.org/tiki-index.php) is an effort to design, implement, and manufacture a free and open 3D graphics chip set and reference graphics card. The OGP was started because existing consumer-level graphics adapters will only work to their full extent with certain specific operating systems, using proprietary drivers. This puts owners of such a card at the mercy of its manufacturer for as long as they are using it, especially on less mainstream, thus less supported, platforms that are left prone to security and maintenance problems.

The Open Hardware Foundation (OHF) came into being as on offshoot of the Open Graphics Project. Unlike software, hardware is a physical item and costs money to produce, lots of money. As an example, the initial run of Open Graphics chips is expected to cost around $2M US to produce, just for the graphics controller chip. To help offset this cost, the project founder, Timothy Miller, started Traversal Technology Inc. Traversal is a for-profit corporation aimed at commercializing and licensing the Open Graphics core.

One of Timothy's concerns in forming Traversal was the company's interaction with the open source community. Because of this concern, Tim suggested the formation of an organization to safeguard the interests of the free/open source community. With Traversal, or any commercial entity interested in making open hardware, such an organization could serve as a guide or reality check, helping the hardware vendor understand the needs and ideals of the people who would buy their hardware.

For these reasons and others, the Open Hardware Foundation was created. The OHF is a non-profit corporation whose stated goals are to facilitate the design, development, and production of free and open hardware. Those of us who formed the OHF believe in free hardware and free software and many of us participate in other open source projects. However, it is much more difficult to turn a hardware idea into a physical device than it is a software idea into a usable program.

One of the key focus areas of the OHF is the production of open hardware. We want to be able to go to our local computer retail store and buy a piece of hardware that is at least an open design and at best a free and open implementation.

The relationship between the OHF and Traversal Technology is one of partnership. Each brings with it key assets. Traversal has IC (integrated circuit) design experience, the desire to make a commercially-successful product, and the desire to contribute back to the community by making that product free and open.

The OHF brings the desire to see a free and open hardware product made available to the average user and to the resources of the free and open source community. The OHF works to bridge the gap between the community developing free and open hardware and the business world producing the hardware.

From a financial perspective the OHF will enable the community to pool its resources to help fund the production of the OGP by providing Traversal a known number of sales. The OHF can then provide the OGP based cards it purchased from Traversal to developers who are working on open source drivers and firmware for the card.

Traversal benefits by having less financial risk associated with producing the graphics chip and the open source community benefits by having hardware available at reduced or no cost for developers who can contribute further to the project.

The other big reason for forming the OHF was to provide an entity to interface with both the development and business communities. Because of the financial requirements involved in semiconductor production, the open source community must work with the business community. For businesses, the ability to work with the open source community gives them access to a wide array of skill sets to which they might not have access. Historically, the two groups have viewed each other with suspicion at best and outright hostility at worst. A major goal of the OHF is to facilitate these two disparate groups working together to produce hardware that is both open and profitable.

Free and open hardware is still in its infancy, though it has been around for many years. However, like free and open software in the 1990s, I believe that free and open hardware is approaching a point of explosive growth potential and that with the right people, the right approach, and the right attitude, that potential can be turned into reality. I, along with all those involved in the OHF and OGP, believe strongly enough that we are committing our time, effort, and financial resources to make it a reality.

Patrick McNamara, President of the OHF, is a Senior IT System Architect for Texas Instruments specializing in configuration management tools and systems. Prior to joining TI, Patrick held positions as senior CM Admin at Nortel and Sabre, as well as software development positions at Sabre and Raytheon Systems. Patrick graduated with a BS in Computer Systems Engineering from the University of Arkansas in 1997. Patrick has a formal background in system level software and hardware development and has had a strong interest in programming and digital logic design as hobbies since childhood. As a member of the Open Graphics Project, Patrick has provided input on the legacy VGA controller design as well as significant contributions to the video controller block.

"We do not yet know the full potential of OCW (OpenCourseWare) and its ultimate impact on global education. But it is clear to us that by thinking of knowledge as a public good for the benefit of all, and acting on this philosophy through OpenCourseWare, we can make a difference."
Susan Hockfield, President of MIT

This article first introduces open content and open educational resources (OER), then compares OER and open source software (OSS), and finally discusses issues of OER project sustainability.

Open Content

Open content is an asset with a structure that is different from that of open source software and open source hardware.

Open content refers to any kind of creative work, such as articles, pictures, audio, and video, or engineering work such as designs, published in a format that is royalty free, share alike and may or may not allow commercial redistribution. Open content explicitly allows the copying and the modifying of the information by anyone. Content can be either in the public domain or under an open license like one of the Creative Commons licenses (http://creativecommons.org/licenses/). The largest open content project is Wikipedia (http://wikipedia.org). The phrase open content was coined to be similar to open source.

Open Educational Resources

Open educational resources (OER) are educational materials and resources that the general public can freely use, distribute and modify without the traditional restrictions imposed by copyright. OER include:

- Learning content: full courses, course materials, content modules, learning objects, collections, and journals
- Tools: software to support the creation, delivery, use and improvement of open learning content including the searching and organization of content, content and learning management systems, content development tools, and on-line learning communities
- Implementation resources: intellectual property licenses to promote open publishing of materials, design-principles, and localization of content

Universities produce and disseminate most OER. Approximately 300 universities maintain more than 3,000 courses online. Content created at universities and then made openly available as OER is frequently not designed or stored for easy sharing and reuse.

MIT OpenCourseWare (http://ocw.mit.edu/) is an initiative of the Massachusetts Institute of Technology (MIT) to make all of its undergraduate- and graduate-level courses online, free and openly available to anyone, anywhere. The MIT OCW initiative encouraged other academic institutions to make their course materials available as OER.

Comparing OER and OSS

OER and OSS are similar in that both rely heavily on sharing materials, publicly accessible repositories of open assets, and licenses that allow the use, modification and redistribution of assets.

OSS relies on collaborative development much more so than OER. With OSS, collaborative development makes the code progressively better. Many eyes decrease the number of bugs in software. However, few OER rely heavily on collaborative development. Two examples of OER that do rely heavily on collaborative development are Curriki (http://www.curriki.org/) and Wikieducator (http://www.wikieducator.org/).

OER and OSS differ in terms of their quality assurance, business models, reuse, and skills required to make changes. OSS strives to be defect free with no errors in the code. There are well established tools and processes that help developers produce defect free software. These tools and processes can not be used to improve open content. The quality of OER is associated with accuracy of facts and the pedagogical methods it supports while the quality of OSS is associated with errors per line of code and the fit between function and customer requirements.

The ways to make money from OER production and distribution are not well understood. Large companies are not making money from OER development. We have a much better understanding on how companies and individuals make money from OSS. Large companies like IBM, SUN and HP invest in OSS projects with the expectation to make money.

File type and pedagogical structure determine the extent in which an OER can be reused, edited or extended. For example, many OER are built so the content is open; however, the file type and structure may make the reuse of the content closed. OSS that runs on one platform but not on others has a similar problem.

Users with no skills in development can make changes to the content of an OER. However, users can't make changes to the OSS unless they have the requisite development skills.

OER Project Sustainability

OER projects involve the production and sharing of OER and the use and reuse of OER by end users.

The OECD defines sustainability of an OER project as the ability of the project to accomplish its goals and continue operations. Sustainability issues are not exclusive to OER. However, what is unique about OER projects is the "determination to give away the results of all these efforts, with no cost recovery mechanisms" (http://tinyurl.com/28wt9h).

The reports 'What makes an Open Education Program Sustainable? The case of Connexions' (http://www.oecd.org/dataoecd/3/6/36781781.pdf) and 'Models for Sustainable Open Educational Resources' (http://www.oecd.org/dataoecd/3/5/36781698.pdf) list some of the ways used to sustain OER projects. These include:

- Endowment: interest generated from the investment of base funding
- Membership: organizations make a one lump sum contribution or annual contributions
- Replacement: funding using proprietary platforms are diverted to fund OER projects
- Foundation: governments or foundations donate money to support the OER project
- Segmentation/conversion: the organization responsible for the OER project provides free content and charges for value added services
- Contributor pays: contributors pay for the cost of maintaining the distribution, while the content provider makes content available for free

OER resources are free for the users but there are technical and monetary requirements that should be covered to be able to produce and share the asset. The cost to maintain OER projects can range from hundreds of thousands of dollars to several millions of dollars per year.

Infrastructure requirements are linked to OER project goals. OER projects require hardware, software, connectivity, human resources, workflow processes, technical support and license policies among other resources.

Most OER projects are either funded by non profit organizations such as the William and Flora Hewlett foundation and the Wellcome Trust or by the universities that established the projects.

Connexions, an OER initiative at Rice University, emphasizes that before considering revenue models for OER projects, the focus of the organizers should be on increasing the aggregate value of the initiative to the users. If users do not perceive value, no revenue model will work in the long term. To provide value for the user, a vibrant OER user community anchored around the OER must exist. One aspect that encourages OER communities is the accessibility to content not only for use, but for modification and distribution.

In his report 'On the Sustainability of Open Educational Resource Initiatives in Higher Education' (http://www.oecd.org/dataoecd/33/9/38645447.pdf), David Wiley explains that there can be many types of interactions between the type of reuse and the publication formats of OER. These interactions will affect the adaptation of OER. The formats that are suitable for publishing might not be the best adaptable by users. Therefore, conflicting goals such as publishing OER efficiently and supporting end-user reuse of OER should find a middle position.

We do not have good ways to measure the health of OER projects yet. Basic metrics such as number of unique visitors and number of downloads are used to assess the health of OER projects.

Conclusions

Several conclusions can be made. OER reside in the public domain or have been released under an intellectual property license that permits their free use or redistribution by anyone. OER and OSS are making a positive contribution to education worldwide. Universities produce most OER, however, these are difficult to reuse.

While OER and OSS share similarities, they have various salient differences. Companies are not making money from OER in the same ways and extent that they make money from OSS. Finally, there are at least six ways to pay for OER

Recommended Resource

FOSS Solutions for OER Summary Report
http://oerwiki.iiep-unesco.org/index.php?title=FOSS_solutions_for_OER_-_summary_report

Monica Mora holds a B.Eng. in Mechanical and Industrial Engineering. In 2004, she became an Assistant Professor at the Technological University of Panama. She has also served as assistant to the president of the university. She is currently a graduate student in Technology and Innovation Management, Faculty of Engineering, Carleton University.

"It may turn out that open source's greatest contribution to organizations is not its great products, but its great working practices. Take a look at where you can take advantage of community within your organization."

Bernard Golden, CEO, Navica

What does it really mean to participate in an open source software community? If a company's open source strategy is limited to acting as end user of open source software, is there a business need to understand the nature of open source communities? Should it be the goal of all businesses to become an active participant in open source communities, or become recognized as significant contributor?

Business users of open source software can broadly be divided into those who use open source software as end-users, and those who incorporate underlying open source technology into their products and services. This article will first address both these groups with the important facets of understanding and evaluating community in the selection of open source software, and then elaborate on the role of active participation in open source communities to enhance the value that can be obtained from the use of open source. It is based upon lessons I've learned from becoming progressively involved in a particular open source software community, Tikiwiki (http://tikiwiki.org), and comparisons with other open source communities which I've made to identify commonalities and differences.

End User Considerations

Many businesses use open source software purely to take advantage of cost savings. However, while the code itself is available at zero cost, total cost of ownership can often escalate due to ongoing support requirements.

Moreover, the initial process of evaluating and selecting open source software can be a time-consuming exercise. Open source software communities spend a fraction of what commercial software companies do on marketing. Product information is therefore often less comprehensive, not centralized in one place, and tends to be more technical in nature. Determining the right open source software to use often requires a fair bit of research. One solution is to hire a consultant who is knowledgeable in open source software for the domain of interest, to do the evaluation.

Understanding the nature of the community that produces the open source software is a critical part of evaluation that is often overlooked by users who are new to open source software. Unlike closed source alternatives, key support options for open source software include non-commercial channels provided by an extended community. Therefore, important criteria when evaluating open source software include the size and vibrancy of the community, the availability of online documentation, and access to support via mailing lists, forums, and IRC (Internet Relay Chat). One source for such information is statistics on sites such as SourceForge (http://sourceforge.net) and Ohloh (http://ohloh.net). However, both sites focus on the contribution of developers and it is important to note that contribution to an open source community is more than just commits to the codebase. As in commercial software, production of good open source software also requires documentation, testing, support, training, and the incorporation of user feedback. An understanding of the maturity of the community can help to answer questions such as "what support mechanisms are available if we roll out this software?" and "how difficult will it be to install and use this software?"

An evaluation of an open source community should also consider the broader ecosystem in which it exists. An environment in which open source is prevalent results in consumer choice that is arguably unparalleled in closed source ecosystems. Open source software has a clear advantage in some domains, for example, in the area of wikis. Freely downloadable open source solutions are plentiful and comparable in terms of quality to their expensive, closed source counterparts.

As an illustration, consider the broad range of needs addressed by two popular open source wiki software, Mediawiki (http://www.mediawiki.org), the platform behind Wikipedia which is excellent for public wikis but not designed with private workspaces in mind, and Tikiwiki which is architected from the start as wiki groupware, making it perfectly suitable for environments that need securing of content for different groups of users. Evaluations for both can be found at Open Source CMS (http://opensourcecms.com), which provides comparative evaluations and comments provided by a large community of users.

Selecting from a final shortlist of software will often require a tradeoff between access to more advanced or specialized functionality, and the size of its community. For example, in an evaluation of content management systems, we find that Tikiwiki, while it supports comprehensive wiki-based collaboration, has a community which is not as large as that of Drupal (http://drupal.org/), an elegant general purpose content management system which lacks wiki functionality.

A deeper understanding of the nature of the community is also critical in determining the nature of support that will be available for an open source software.

It is important to consider the nature of the resources the community has to offer in relation to the capabilities of your organization. For example, the Tikiwiki and Drupal communities have a large number of highly technical members who can provide support at a high level of technical sophistication. On the other hand, Joomla (http://joomla.org/), has a relatively smaller team of technical experts combined with a larger community of less technical but more design oriented individuals such as graphic and web designers. Consequently, Joomla support forums tend towards questions of a "how-to" nature and discussions over more complex technical issues are less prominent. On the other hand, it is easier to find consultants for low cost design customizations for Joomla than it is for Drupal or Tikiwiki. Another benefit of open source software like Joomla that is exposed to a wider swath of mainstream users is that they tend to be more user-friendly, although this usually comes at a price of reduced functionality.

Is Modification Required?

When evaluating open source software, a careful analysis of features is needed to determine if the software is suitable as-is or if modifications will be required. If the need for modifications is likely, a further analysis of the core architecture as well as the project itself is beneficial. This is especially important for businesses that intend to use open source software as underlying technology for their products or services. As an example, consider the differences between Drupal and TikiWiki.

In terms of architecture, Drupal has a smaller core which provides a set of hooks at a lower level; these can be leveraged when creating custom components that provide "core-type" functionality.

Tikiwiki has hooks at a higher level that are used mainly by add-on components that enhance user input, output display, support new content types, and provide integration with third party systems. To add substantial "core-type" functionality in Tikiwiki, one would have to actively participate in the Tikiwiki development team; whereas in Drupal it is possible to lead and maintain the development of substantially sized new components without direct involvement with the Drupal core team. In making a final platform decision, it is therefore necessary to evaluate if development and maintaining of new components is needed, while taking into account the scope and nature of the effort involved.

Businesses incorporating open source software into their products may want to consider leading the development and ongoing maintenance of a new component. This could lead to benefits that come through influential leadership of a new sub-community which can provide ongoing support and resources. The larger the potential sub-community, the greater the benefits, but greater also are the risks and the costs.

There is also a question whether demand for that component will grow sufficiently to make it important enough to the community as a whole. Creating a component, as in the case for Drupal, has to be weighed against the alternative of participating actively within an established group of developers, such as in Tikiwiki, where the component is already part of its core. While dealing directly with the core development team relinquishes leadership, it can still enhance one's role in the community through active participation.

When working with open source software, it is important to consider the roadmap of the community and evaluate if it matches one's desired use and objectives. Unlike closed source software where the product roadmap is defined by management, the roadmap of open source software is in a constant state of flux influenced by the community of users, developers, and other contributors. Understanding where a community is headed requires a perceptive evaluation of the individual motivations of the various stakeholders involved.

Benefits of Collaboration as End User

Even if business use means using software as-is with no plan to develop or contribute code back to the community, there is a strong business case for some collaboration with the open source communities for the software the business is using. Implementing open source software can require time spent researching existing documentation and asking for help on forums. In open source, the nature of problem solving is highly interactive and collaborative. Large projects often have community members that respond to requests for help on IRC. One should keep in mind that support is often provided by a group of unpaid volunteers, each with their own schedules, and should not be surprised that there can be no answer at times, and many people rushing to help at other times.

Someone unfamiliar to open source might wonder why people are willing to volunteer their time. Responding to support questions gives community members a feel for what aspects of the software can be improved and helps to prioritize feature enhancements. Moreover, many participants in open source communities make their livelihood from the software that community produces.

They understand that a strong community leads to the strength of their individual consulting, hosting, or product businesses. In addition, providing support leads to more users which results in more testing of the software, in turn improving the product.

In the course of using either open or closed source software, bugs are often detected. In open source, bugs and feature requests are typically submitted through the community's bug tracker software so they can be acted upon by the community. Being actively involved in the reporting of new bugs and feature requests raises your profile in the community, and increases the likelihood of receiving help and requests for additional comments or suggestions in future.

For users who are also developers, it is often easier to fix the bug or code the feature enhancement oneself rather than to wait for someone else. By sharing these changes with the rest of the community, others can benefit and the changes can be integrated into the main stream of development. Sharing modifications benefits not just the community but also the contributor who subsequently benefits from ongoing testing and support of the code by the rest of the community.

Finally, geographical location is useful contextual information to otherwise impersonal online communication. It is useful to visit key project members in your vicinity, especially if you are intending on being actively involved in a community. Meeting face to face can add a personal touch to an otherwise staid relationship.

Understanding Community Differences

Businesses who intend to participate actively through contributing code back to the community, especially those whose products depend on the open source software, will have to gain an appreciation of the different cultures that exist in each community in order to maintain a cordial working relationship as participation becomes progressively involved. Open source communities may become suspicious of corporate intentions if they are perceived as being in conflict with the needs of existing members.

Our experience is that it is best to be as up front as possible about one's plans, especially with key members of the community. Most open source communities are characterized by more open policy discussions using tools like wikis, forums, and mailing lists than is typical in corporate environments. It is also important to be aware of the software development management procedures that are in place in the community, many of which are likely to differ from those used by your organization.

For example, different communities have different standards as to who can be a code committer. Some communities have strict published guidelines while others are more flexible. In the Tikiwiki community, any user that has contributed at least one code patch of decent quality is typically approached and encouraged to contribute their changes directly. As such, the Tikiwiki community provides a relatively large number of its developers with direct commit access to its revision management system.

Developers who are more familiar with corporate closed sourced development environments may find the looser control over code commits unusual, and worry about lack of control over changes to the codebase. However, the "wiki-way" characterized by open collaborative authoring, when applied to software development is surprisingly effective.

Tikiwiki's experience has been that most new developers are extra careful with their commits, since no one wants to get a bad reputation for submitting shoddy work. Moreover, before a developer is given commit access, he is directed to documentation that details expectations regarding coding conventions and practices. Any developer that is not confident in meeting those requirements is likely to avoid accepting commit access.

Core developers keep a close watch on new commits; gentle warnings as well as requests for clarification are not uncommon. This creates an environment of rapid innovation driven by quick feedback and discussion between collaborators. At times, changes to the code are rolled back by core developers followed by a gentle "what is this commit trying to achieve?" or "how about trying something else instead?" Such exchanges are instructional and lead to unexpected innovations, much more so than having discussions on paper. Nevertheless, design and architectural discussions are necessary when significant changes are planned. These are often conducted first over IRC and then documented on wiki pages and forums so that the rest of the community can comment.

Many projects have an editorial board or a documentation team, providing a method for non-developers to participate. Like the Tikiwiki developers, the Tikiwiki editorial board conducts extended virtual meetings using wiki pages, forums, IRC, and the mailing list. The mix of synchronous and asynchronous mediums of communication help to overcome the timezone differences faced by this diverse group.

It is often possible to collaborate with other community members. In any community there are complementary needs to be discovered. For each individual, the community provides a ready pool of resources that help to smooth out demand fluctuations that each face in their own businesses. Different open source communities have different cultures and varying levels of institutionalization for monetary transactions between community members. In some communities, privately arranged monetary transactions for work done between members are common, although there is no officially sanctioned bounty system. In other communities, such as GNOME, official bounties are often provided by the community for certain features. In almost every community, contributing code created as part of paid projects back to the community is strongly encouraged.

The licensing of open source software can provide a clue to the expected level of code sharing. The LGPL (Lesser GNU Public License) used by Tikiwiki suggests a lower expectation than communities of software licensed under the GPL (GNU Public License), while those licensed under academic style licenses such as the BSD (Berkeley Software Distribution) license are characterized by an even lesser expectation. Businesses who are more familiar with commercial software development should refrain from the knee-jerk reaction to pay members of the community to solve every problem. Research has found that paid work within open source communities can lead to crowding out of intrinsic motivations to contribute (http://www.slideshare.net/nice/crowding-effects-how-money-influences-open-source-projects-and-its-contributors/).

Rising to Leadership Positions of Influence

Companies that depend on open source software as components for their products or services should aim to rise to leadership positions of influence within the communities they participate in. The path to leadership involves initiating conversations with users who have not previously contributed to encourage them to be more involved, depending on the goals of the contributor and the amount of time available to work on related items. Members with ideas, patches or documentation with no time to integrate them can be introduced to other contributors so they can collaborative on getting more resources into the community.

Open source communities often hold regular events to work on new software features together. These are excellent opportunities to interact and to get to know fellow community members better. Events are also organized in preparation for the release of a new version of the software to fix outstanding bugs in order to accelerate the release of the next version. Members contributing to documentation usually take this opportunity to refine the manuals, online help, and the documentation website.

By contributing to these events, a business can raise its profile within the community and increase the familiarity and comfort level other members have of its participation. A business should also take every opportunity in the marketing of its own products to support and champion the open source community, whether it is in product marketing, corporate communications, or industry forums. Such efforts are often greatly appreciated and reciprocated.

Eventually, active participation and support for the community means being offered opportunities to take on official leadership responsibilities including managing key software components, organizing events, or coordinating software releases and documentation.

Conclusion

Understanding the community which develops the software being used by a business provides many benefits. As an end user, the company is able to access support channels traditionally not available with closed software solutions. As an active participant, the company has the opportunity to influence the direction of the software, and if desired, to leverage that influence as part of its business strategy.

Additional Resources

Enterprise Open Source Directory:
http://www.eosdirectory.com/

Business Readiness Rating:
http://www.openbrr.org

Nelson Ko is the founder and CEO of Citadel Rock Online Communities Inc. (http://citadelrock.com), providing solutions for online collaboration using wikis, social networking and multimedia messaging. He is an active contributor to the Tikiwiki open source project. Nelson has held positions in Hewlett-Packard and Singapore Telecom, and architected solutions brought to market across the world for companies such as Trans World International Interactive and Telstra. He holds an M.A. Economics degree from the University of Toronto, and is currently working on a dissertation in the M.A.Sc. Technology Innovation Management program at Carleton University.

Q. What is the current state of open source in public administration?

A. Public administrations across the globe are increasingly turning to Free/Libre Open Source Software (F/LOSS). Vendor lock-in, cost control and an overall quest to gain back the control of their Information Technology (IT) are just a few of the reasons attracting them to open source operating systems and thousands of open source applications. We know about flagship projects such as the city of Munich in Germany and the Spanish state of Extremadura (http://ec.europa.eu/idabc/en/document/1637/470). However, many more projects are happening. France is very active with the Finance Ministry and National Police leading the way.

More or less every country today is slowly integrating F/LOSS. In fact, the European Union produced a very complete 287 page report (http://flossimpact.eu/) last November, stating that open source is beneficial to European businesses and that its usage increases productivity. Another interesting fact is that in Europe the number of feasibility studies has decreased, a sign that the time for studies is over and implementations are under way. We can see the same trend with the Government of the Province of Quebec as being a few years behind Europe but with the same types of projects taking place.

So should the next step be full adoption of open source in all government departments with mandatory requirements in all request for offers? Not exactly. This paradigm transition is a major one as the computer industry is reaching adulthood. Like the construction industry a century ago, IT will now have to provide its users with standard interoperable components with full ownership and transparency.

And, like any major change, it will take some time to take place. No organization would want to upset its current suppliers; these suppliers, today's computer industry, are also living this major paradigm switch and some are only starting to realize the major impact that it will have on them.

With the number of legacy software in use in government today, it would be unwise for a public administration to upset the vendors supporting the vast majority of their install base. Not to mention the fact that many projects are just as much linked to the people as the software used. Many of these people are business process experts and the value of their knowhow exceeds the value of the tools used. So care must be taken to ensure continuity.

Providers of F/LOSS services should also not wish to have too quick of a transition. Imagine a large government department with thousands of users wanting to switch everything to free software tomorrow. Can we deliver? Do we have a solution for everything? How will change management and user acceptance affect the success of our projects?

It is best to go slowly and start migrating one part at a time. Also, let's not forget, the paradigm has changed: increasingly, clients will migrate themselves without the need for outside help. In fact, I predict that the most used method will be to call on temporary help to increase their IT team and do knowledge transfer. So governments are switching, they will not and should not do it quickly, and they will do it differently by looking for temporary help and knowledge transfer instead of old style contracts.

We know what is coming, we have the luck to be behind here in Canada so we can learn from other parts of the world and prepare service offerings to meet the demand of governments. In the words of Andrew Clunis of the One Laptop Per Child Project: "Our lagging adoption of free software provides us with the opportunity to learn from the other nations' experience."

Christian Meloche has over 20 years of experience in Information Technology. He is Vice President of Operations for Infoglobe (http://www.ottawa.infoglobe.ca). He has been International Operations Director, Manager of Information Systems, Project Leader, Network Administrator, Analyst and Programmer. He has worked for Foreign Affairs Canada, Netscape Communications Corporations, AOL & Time Warner. He can be reached at cmeloche@infoglobe.ca .

Q. Do venture capital firms invest in open source companies in Canada and the US?

A. Venture capital (VC) firms have invested approximately $1.9 billion in open source companies in the United States since 2001. No venture capital firm has invested in an open source company in Canada.

The definition of what constitutes an open source company complicates the meaning of the numbers somewhat. For instance, do the numbers include pure-play open source companies or do they incorporate traditional vendors that have released some proprietary code to open source communities?

The upward tracking of VC investments in open source companies contrasts with the general downward slide of venture funding of traditional software vendors over the same period.

Robin Vasan, Managing Director with top-tier US-based Mayfield Fund and investor in open source businesses, was quoted in the Computer Business Review Online earlier this year on the topic: “VCs are not too excited about traditional enterprise software companies anymore: people don’t want to buy business enterprise software the way they bought enterprise software anymore”. He further adds: “Open source is not a market, it is a new form of development and distribution. When you look at it that way then the rationale as to why so much money has gone into open source makes more sense”.

Matt Asay, open source industry veteran and founder of the Open Source Business Conference observes that savvy VCs who get the power of open source are placing bets on promising new ventures as well as established traditional companies migrating towards OS offerings.

There have been several notable VC investments in US open source companies. These include Alfresco ($10M), JasperSoft ($23.5M), Pentaho ($13M), Vyatta ($18.5M), and Zimbra ($30.5M). Unlike recent bubble-like VC activity in Web 2.0 companies, investors funding open source ventures appear to be paying more attention to the realism of the underpinning business models in addition to keeping a close eye on the prospects for a successful exit.

And the good news is that exits are happening. Some of the notable and recent open source initial public offerings (IPOs) and mergers and acquisitions (M&As) include Sourcefire which raised $71.8M in 2007, GlueCode Software which represented IBM's first acquisition of an open source company, and JotSpot which was acquired by Google.

So what is happening in Canada? To say that locating comparable data on venture funding in Canada or Ontario of open source companies is difficult is an understatement. There is virtually little information on the state of Canadian VC investment in open source software companies. Further, portfolio holdings of Canadian-based VC firms reveal no readily identifiable investments in open source companies. Three points can be made:

1. There are too few open source companies in Canada

2. Regardless of the actual number of open source companies operating in Canada, none or few of them are attracting or wish to attract VC funding

3. Canadian VC firms do not have the same experience with open source investments as do VC firms in the US

It would be difficult to imagine Canadian software entrepreneurs not taking notice of open source technology as a means to innovate and compete. The August 2007 issue of the Open Source Business Resource with its focus on open source business models reveals that in fact there are an impressive number of entrepreneurs leveraging the power of open source in Ottawa alone. It would not be a stretch to imagine the same phenomenon occurring in other major Canadian cities. Thus, point one above, though difficult to quantify, is probably unlikely.

Point two, on the other hand, is relatively easy to verify. Diverse sources which track venture investments in a broad range of industry sectors make it apparent that there is virtually no VC funding of open source companies in Canada. Further research based on US sources again produces no information on venture deals of Canadian-based open source ventures.

Is it fair to assume that Canadian VCs shun investing in software companies that have chosen to grow their businesses via open source as in point three? Or is it more plausible that Canadian VCs are just more cautious and are taking a “wait-and-see” approach relative to their US counterparts? Perhaps there are simply not enough investment-grade open source ventures that merit VC attention in this country? Whichever is correct, will this lack of VC funding in Canada matter in the long run for domestic open source software companies? Only time will tell.

Luc Lalande is the current Director of the Innovation Transfer Office at Carleton University. Since joining Carleton University in 1996, Luc has initiated and helped implement a number of programs and events aimed at stimulating technological entrepreneurship including the Social Innovation Challenge, Foundry Global and the OttawaTechWiki project. In recognition of his efforts, Luc was honoured to receive the Des Cunningham Award in April 2002 from OCRI. The award is presented to an individual who had made a significant contribution to forging business-education partnerships or facilitating government-industry interaction. Luc often cites the Ottawa high-technology community's tremendous goodwill and willingness to share knowledge as the principal reasons for the success of the initiatives he has helped launched at Carleton.

FOSS Open Content Primer

Published and Copyrighted by: United Nations Development Programme - Asia-Pacific Development Information Programme

From the Description:

In the move towards an information era, the question of systems designs becomes a critical one. The dominant model of treating information as property through Intellectual Property regimes such as patent and copyright has proved to be detrimental to a number of developing countries. It is now recognised the a very stringent copyright regime threatens free speech, creativity and access. The Open Content Movement, (inspired by the free software movement) has slowly gained momentum as a serious alternative to the exclusionary world of copyright. This primer looks into Open Content is a model of that offers a different ethic of production and distribution, and as an important model through which knowledge and information can be democratized, especially, but not limited to developing countries.

http://www.iosn.net/open-content/foss-open-content-primer

Current Status of F/OSS

Copyright: Guadalupe Morgado, Manon Van Leeuwen, et al.

From the Introduction:

This work aims to reveal results of a survey run by the tOSSad (http://www.tossad.org/) project. This particular study has been devised and conducted to investigate further administrative issues and obstacles against F/OSS adoption. Those issues intermingle with financial, technical, legal, and personal factors. Therefore the majority of survey variables devised to capture perception of public administrators around Europe regarding the importance they attach to the factors such as F/OSS product quality, availability of support, expertise and documentation, TCO, vendor lock-in, political influence, administrative attitudes, productivity, and training costs. Based on their F/OSS experience and knowledge so far the survey respondents are asked to rate each of those factors separately as a barrier against wider F/OSS adoption in public administration.

http://www.tossad.org/content/download/1385/6894/file/tOSSad_D18_V2.3.pdf

Adapted Consulting Inc. Launches OpenFM

August 22, Toronto, ON

When Adapted's Co-Founder, Frédéric Renet, couldn't find a commercial radio that was right for this situation – durable, energy efficient, easy to repair and affordable – he decided to build his own. From this first design created to withstand the harsh climate of Mali comes Adapted's OpenFM. OpenFM is a low power, affordable open source FM radio station created for use in remote areas. The system has been designed with harsh weather conditions in mind; it is able to withstand heat, dust and humidity with little manual intervention. It functions on solar power and so is particularly suited for rural environments that lack dependable electricity. OpenFM consists of a low power FM transmitter; antenna; low bandwidth, low power computer; mixing board; and open source audio software. The open source component of the OpenFM is perhaps the station's most innovative attribute. The design has been registered using the creative commons license, which means that the blueprint of the device is available to those who wish to create their own OpenFM or amend the equipment.

Press Release: http://adaptedconsulting.com/files/OpenFM-PressRelease.pdf

OpenFM Wiki: http://openfm.adaptedconsulting.com/index.php/Main_Page

RBC Financial Group Launches Student p2p Forum

August 22, Toronto, ON

RBC p2p (http://www.rbcp2p.com/), an online forum where students can speak openly and honestly about their financial struggles and successes - is scheduled to go live by the end of August. The site's content will be driven by students, while the six new student hires will manage discussions, blog about their own banking experiences, and share information and solutions about the money issues that matter most to undergrads and graduate students.

Open Access to Health Research

September 4, Ottawa, ON

Today, the Canadian Institutes of Health Research (CIHR) unveiled a new policy to promote public access to the results of research it has funded. CIHR will require its researchers to ensure that their original research articles are freely available online within six months of publication. "Timely and unrestricted access to research findings is a defining feature of science, and is essential for advancing knowledge and accelerating our understanding of human health and disease," stated Dr. Alan Bernstein, President of the Canadian Institutes of Health Research. "With the development of the internet it is now feasible to disseminate globally and easily the results of research that we fund. As a publicly-funded organization, we have a responsibility to ensure that new advances in health research are available to those who need it and can use it - researchers world-wide, the public and policy makers."

http://www.cihr.ca/e/34851.html

September 20

Break Free Seminar

Toronto, ON

SQL POWER is hosting a free Breakfast Seminar on the topic of Open Source BI Tools and how to Break Free from Proprietary BI. The purpose of this Seminar is to educate IT Consultants, Managers, Directors and CIOs on the benefits of open source technologies, how to introduce these technologies to their respective organizations, and most importantly how to utilize open source tools to free up the BI budget. The guest speaker, Andre Boisvert, is the Chairman of Pentaho Corp. and is one of the leading authorities on Business Intelligence and open source software in North America.

http://www.sqlpower.ca/page/breakfreeseminar

September 24-25

Connections 2007

Toronto, ON

NRC believes that creating globally competitive technology clusters is one of the best strategies for fostering a nation's economic growth. Our intention is to bring key players from communities across Canada together for two days of dialogue, exchange, problem-solving and networking. We hope you will leave NRC Connections 2007 armed with information and tools that will help your cluster move to the next level. Key themes will include: Innovating to Succeed, Working Together, Building Networks and Links, and Branding and Marketing Clusters.

http://connections-connexions2007.nrc-cnrc.gc.ca/welcome_e.html

---

September 24-27

FOSS4G 2007

Victoria, B.C.

The 2007 Free and Open Source Software for Geospatial (FOSS4G) conference gathers developers and users of open source geo-spatial software from around the world to discuss new directions, exciting implementations, and growing business opportunities in the field of open source geo-spatial software. Focused on the practical "make it work, get it done" world of open source application development, this annual conference boasts a very high concentration of geo-spatial technical opinion leaders. Attendance at this event has grown at over 50% a year since its inception in 2003, paralleling the rapid growth and adoption curve of open source geo-spatial tools in the marketplace.

http://www.foss4g2007.org

September 27

Canadian Geospatial Data Infrastructure Project Showcase

Calgary, AB

GeoConnections invites you to attend the first of a series of Canadian Geospatial Data Infrastructure (CGDI) Project Showcases. The event is targeted at past, current, and future CGDI end-users from all levels of government, non-government organizations, the private sector, Aboriginal groups, and academia. The CGDI Project Showcase will help users become more aware of the possible uses of the CGDI to support decision making in their areas of interest. In addition, the Project Showcase will create a network of CGDI users who will continue to learn from each other as they evolve their decision-support systems. Attendees will also gain knowledge of the various GeoConnections funding opportunities.

http://www.geoconnections.org/en/newsmedia/whatsnew/getDoc=793

September 27-28

ICEG 2007

Montreal, PQ

The International Conference on e-Government (ICEG 2007) invites researchers, practitioners and academics to present their research findings, work in progress and conceptual advances in any branch of e-Government. The meeting brings together varied groups of people with different perspectives together into one location, for the purposes of helping practitioners find ways to put research into practice, and for researchers to gain an understanding of additional real-world problems. The conference includes a mini-track on Mini track on Software and Interoperability issues in e-Government (open source software).

http://www.academic-conferences.org/iceg/iceg2007/iceg07-home.htm

---

October 13

Ontario Linux Fest 2007

Toronto, ON

Ontario Linux Fest is a conference designed to present compelling topics of interest to users of Linux and open source software. These topics span a range of interests from technical to motivational, educational to organizational and social to legal. Attendees will find out what is happening in the open source world from the people directly involved. It's a great event to catch up with old friends, meet project contributors and develop new business relationships.

http://onlinux.ca/

October 15-17

GTEC2007

Ottawa, ON

The GTEC Conference attracts the senior vanguard of IT decision makers from across Canada and around the world. The GTEC conference tracks will be a unique forum for discussing Government Policy Initiatives, Trends in Program Management, for exploring Emerging Technologies and discussing the challenges governments face in Shared Infrastructure and Solutions. Over an engaging 3-day conference, we will explore the dynamic business environments that are being driven by web 2.0 internet applications and solutions. We will discuss how the evolution of internet-based technologies is driving the "business of government" from "government 1.0" to "government 2.0".

http://www.gtec.ca/conference/conference.html

October 17

Building a Better Mouse Trap

Ottawa, ON

This is the second in the eBsuiness cluster series of events dedicated to Ottawa area Start-ups. This event will focus on the supply side of commercialization or “how to build a better mouse trap”. For this event we have drawn on the experiences of Tobi Lutke chief technology instigator at Jadedpixel and Michael Weiss professor at Carleton University. Both presenters offer unique start-up perspectives from the Academic to world class implementation. Please join us for insights into excellence and how to build a better mouse trap.

http://www.ebusinesscluster.com/

October 21-23

WS2007

Montreal, QC

The 2007 International Symposium on Wikis brings together wiki researchers, practitioners, and users. The goal of the symposium is to explore and extend our growing community. The symposium has a rigorously reviewed research paper track as well as plenty of space for practitioner reports, demonstrations, and discussions. Anyone who is involved in using, researching, or developing wikis is invited to WikiSym 2007! We recognize the online world is always evolving, and we also welcome contributions which are about other online media consistent with the wiki philosophy of being open, organic and participatory.

http://www.wikisym.org/ws2007/index.html

October 22

Workshop on Integration of Open Source Components into Large Software Systems (Co--located with OOPSLA 2007)

Montreal, QC

Developing large software systems has largely become an exercise in integration. About 85% of code that goes into the software of a typical system is written by others, and the main role of businesses is to write the glue that holds the externally developed components together. While in the past, businesses were largely concerned with the integration of commercial off-the-shelf (COTS) components, many of these components will now come as free/open source software (F/OSS) components. The use of open source components provides new strategic options for reducing the exposure to risk and cost of development, while significantly increasing the available solutions. Models for the integration of COTS components do not necessarily apply to open source components. A particular focus in this workshop will be on the shift away from COTS to F/OSS components, and what new opportunities and issues are introduced by it.

http://www.carleton.ca/tim/oopsla

October 25-26

FSOSS07

Toronto, ON

FSOSS is a high-profile event that attracts leaders from industry and the open source community in order to discuss open source issues, learn new technologies, and promote the use of free and open source software. The Symposium is a two-day event aimed at bringing together educators, developers and other interested parties to discuss common free software and open source issues, learn new technologies and to promote the use of free and open source software. At Seneca College, we think free and open source software are real alternatives.

http://fsoss.senecac.on.ca/2007/

The goal of the Open Source Business Resource is to provide quality and insightful content regarding the issues relevant to the development and commercialization of open source assets. We believe the best way to achieve this goal is through the contributions and feedback from experts within the business and open source communities.

OSBR readers are looking for practical ideas they can apply within their own organizations. They also appreciate a thorough exploration of the issues and emerging trends surrounding the business of open source. If you are considering contributing an article, start by asking yourself:

1. Does my research or experience provide any new insights or perspectives?

2. Do I often find myself having to explain this topic when I meet people as they are unaware of its relevance?

3. Do I believe that I could have saved myself time, money, and frustration if someone had explained to me the issues surrounding this topic?

4. Am I constantly correcting misconceptions regarding this topic?

5. Am I considered to be an expert in this field? For example, do I present my research or experience at conferences?

If your answer is "yes" to any of these questions, your topic is probably of interest to OSBR readers.

When writing your article, keep the following points in mind:

1. Thoroughly examine the topic; don't leave the reader wishing for more.

2. Know your central theme and stick to it.

3. Demonstrate your depth of understanding for the topic, and that you have considered its benefits, possible outcomes, and applicability.

4. Write in third-person formal style.

These guidelines should assist in the process of translating your expertise into a focused article which adds to the knowledgable resources available through the OSBR.

Upcoming Editorial Themes

October 2007	Open Source Licensing
November 2007	Support
December 2007	Clean IP
January 2008	Interoperability
February 2008	Data
March 2008	Procurement

Formatting Guidelines:

All contributions are to be submitted in .txt or .rtf format and match the following length guidelines. Formatting should be limited to bolded and italicized text. Formatting is optional and may be edited to match the rest of the publication. Include your email address and daytime phone number should the editor need to contact you regarding your submission. Indicate if your submission has been previously published elsewhere.

Articles: Do not submit articles shorter than 1500 words or longer than 3000 words. If this is your first article, include a 50-75 word biography introducing yourself. Articles should begin with a thought-provoking quotation that matches the spirit of the article. Research the source of your quotation in order to provide proper attribution.

Interviews: Interviews tend to be between 1-2 pages long or 500-1000 words. Include a 50-75 word biography for both the interviewer and each of the interviewee(s).

Newsbytes: Newsbytes should be short and pithy--providing enough information to gain the reader's interest as well as a reference to additional information such as a press release or website. 100-300 words is usually sufficient.

Events: Events should include the date, location, a short description, and the URL for further information. Due to the monthly publication schedule, events should be sent at least 6-8 weeks in advance.

Questions and Feedback: These can range anywhere between a one sentence question up to a 500 word letter to the editor style of feedback. Include a sentence or two introducing yourself.