

Managing the Procurement, Use and Distribution of Free and Open Source Software

**Eric J. Smith
Fraser Milner Casgrain LLP
Ottawa, ON**

**Canadian IT Law Association Spring Training Conference
April 30th, 2009
Toronto, ON**

Introduction

The development and wide-spread adoption of the Internet (including the World Wide Web) has fundamentally altered the way in which we interact with one another as well as the way in which we perform many tasks. Whether finding the latest news, shopping, or communicating with our friends and colleagues, the Internet has become a key tool in our everyday lives.

The Internet has also had a profound effect on the way in which computer software is developed and distributed. Prior to the widespread use of the Internet, individuals typically developed software either alone or in relatively small groups. The lack of a means to easily communicate with other developers made widespread collaboration impractical. In addition, it meant that computer software was most often distributed on tangible storage media, such as floppy disks and later compact discs.

The Internet provides a fast and efficient means by which developers, regardless of where they reside or who they work for, can collaborate with one another on development projects. The ability to collaborate and share code “on-line” has been a boon to the popularity of free and open source software, as such software can be easily located and obtained for use by others, whether for individual use, within an organization’s IT infrastructure, or for redistribution “as is” or in modified form.

With the advent of this readily available source of software, corporations and other organizations have found themselves playing catch-up in their attempts to understand the

benefits and risks of using free and open source software and how to manage their use. This paper provides a brief overview of some of the key issues associated with the use and distribution of free and open source software as well recommendations for managing such use.

Free and Open Source Software

While most people involved in the technology industry are familiar with the concepts of “free software” and “open source software”¹ (hereinafter referred to as “FOSS”), a brief overview is necessary for the uninitiated. Though the sharing of source code was not uncommon in the early days of computer programming, once organizations began to recognize the value of their computer programs, they increasingly restricted access to the source code for their programs. To the extent they made copies of the program available to the public, such copies contained only the machine-readable binary code. The use of such programs was also subject to the terms and conditions of a software license which typically permitted only limited use of the software, and prohibited users from making modifications to the software (which is difficult to do without access to the source code) or reverse engineering or decompiling the software in order to discover its source code or logical structure. This distribution model, which remains prevalent today, allows the software vendor to control the reproduction, use, modification and redistribution of its software products. As such, licensees are dependent on the vendor for bug fixes and improvements to the software.

¹ The labels “free software” and “open source software” each have their advocates and though they can most often be applied to the same types of software, the term “free software” emphasizes the rights or freedoms that individuals have to use, study, modify and redistribute code, while the term “open source software” focuses more on the collaborative development model which relies on the availability of source code to facilitate the study and improvement of code. See <http://www.gnu.org/bulletins/bull1.txt>, <http://www.gnu.org/philosophy/free-sw.html> and <http://www.gnu.org/philosophy/open-source-misses-the-point.html> for more information about the term “free software.” See “Goodbye, “free software”; hello, “open source””, by Eric S.Raymond, <http://www.catb.org/~esr/open-source.html>, <http://www.opensource.org/about> and <http://www.opensource.org/history> for more information about the term “open source software”.

In the case of FOSS, the source code to the software is made available and recipients are free to use, modify and redistribute the software. Unfortunately, many people continue to hold the mistaken belief that FOSS can be used and distributed without restriction or conditions. However, just as non-FOSS is protected by copyright (and is also sometimes subject to patent rights), so too is FOSS, and while the terms and conditions of the licenses under which FOSS is distributed are generally more liberal than the licenses under which non-FOSS is distributed, users of FOSS must still comply with their terms.

While it is beyond the scope of this paper to provide an in-depth analysis of each type of FOSS license, it is important to note that not all FOSS licenses are the same. While generalizations obscure the fact that there are many different FOSS licenses, a brief discussion of two major subgroups of FOSS licenses will illustrate some of the common conditions found in different types of FOSS licenses.

(a) Academic Licenses

Examples of academic licenses are the MIT (X) License, the BSD License, the Apache License and the Academic Free License.² These licenses grant to any person obtaining a copy of the software to which the license applies a license to use, copy and redistribute the software, without or without modification. Generally speaking, the primary obligations imposed on the recipient who wishes to redistribute copies of the software, with or without modifications, is that the recipient reproduce the copyright and the permission (or grant of license) notice, including disclaimers and other conditions, in all copies distributed. Some academic licenses impose additional conditions, such as the notorious “advertising” clause that forms a part of the original

² Though it is common practice to refer to licenses by names such as “BSD License” or “BSD-type license”, it is important when considering the use of FOSS that one reviews the exact license(s) found within the code. Similarly labeled, or different versions of the same license can contain important differences.

UCB/LBL version of the BSD license³, and prohibitions on using the names of organizations or individuals holding the copyright or involved in the development of the code to endorse or promote products derived from the software.

While the MIT (or X) and BSD licenses and their derivatives are relatively short and informal licenses, the Apache (Version 2.0) and Academic Free License are examples of academic licenses to which a more formal legal discipline have been applied. For example, version 2.0 of the Apache License provides definitions of key terms found in the license, it provides an express grant of license to patent claims of “Contributors” that are embodied in the program and includes a patent retaliation clause which stipulates that if any licensee of the program institutes a patent infringement claim against another user of the program alleging that their use of the program infringes their patent rights, the patent licenses granted to the licensee instigating such suit shall terminate.

Equally important, version 2.0 of the Apache license expressly states that a licensee who modifies the Apache v.2.0-licensed code or creates “Derivative Work” based on such code, can license its modifications or Derivative Works under additional, or different, terms and conditions, provided such distribution otherwise complies with the terms of the Apache version 2.0 license. As such, the creator of a Derivative Work is not required to make the source code to such Derivative Work available to others, or to license the Derivative Work under the terms of a FOSS license.

As the brief overview above illustrates, software licensed under academic licenses may be incorporated, with or without modifications, into non-FOSS applications and licensed under the terms and conditions of a non-FOSS license. This makes software distributed under

³ The license was amended in 1999 to remove such clause and the University of California rescinded the clause so that it no longer applies to software in which it found.

academic licenses particularly attractive to organizations that wish to receive the benefits of FOSS, but do not want to make the source code to their software available to others. However, even if an organization is not required to make the source code to their software available, as noted above, there are obligations that it must fulfill in order to comply with the terms of the license under which the FOSS was distributed.

(b) Reciprocal Licenses

The other major sub-group of FOSS licenses are often referred to as “reciprocal” or “copyleft” licenses. The GNU General Public License (also referred to as the GNU GPL or just simply the GPL)⁴, published by The Free Software Foundation is probably the best known example of a license that uses the concept of “copyleft” in order to preserve the freedom of recipients of FOSS to use, study, reproduce, modify and distribute such code.

Under the concept of “copyleft”, the holder of the copyright in a computer program grants licenses to use, copy and modify the work, and distribute copies of the work (with or without modifications) so long as all copies of the work and modifications made to the original work are distributed under the same terms and conditions. Such requirement, together with other conditions such as the requirement to ensure that source code and other materials necessary to allow others to modify the code are made available, ensure that a recipient of GPL-licensed code cannot take advantage of the benefits of free software (e.g. to make modifications to the code) and then place restrictions on others to whom it distributes copies of the software (e.g. licensing the modified code under the terms of different license which does not allow recipients to make

⁴ Version 1 of the GNU GPL was released in January 1989. Version 2 of the GNU GPL was released in June 1991, along with the release of the Library General Public License (LGPL) (which, though it was the first publication of such license, was also numbered version 2 so that it would have the same version reference as GNU GPL version 2). On June 29, 2007, version 3 of the GNU GPL was released. For a detailed discussion of version 3 of the GNU GPL, see [GNU General Public License: An Addition to the Free Software License Genus](#) a copy of which can be found at www.fmc-law.com. A copy of the GNU GPL can be found at <http://www.gnu.org/licenses/licenses.html>.

modifications, or restricting access to the source code for such modifications). The GPL also goes further than some other reciprocal licenses in that it provides that you may not distribute works that include or are derived from or based on GPL-licensed code, in whole or in part, unless the entire work is distributed under the terms of the GPL, including the requirement that source code for the entire work be made available. It is for this reason that the GPL is often referred to as a “viral” license.

Other examples of reciprocal licenses include the Mozilla Public License (“MPL”) and the GNU Lesser Public License (“LGPL”). Unlike the GPL, the MPL applies the reciprocal licensing requirement only to the distribution of the original code licensed under such license and modifications made to it by the licensee. If MPL-licensed code is incorporated into a larger work, only the MPL-licensed code, and any modifications made to such code, must be distributed under the terms of the MPL. The remainder of the larger work can be licensed under terms and conditions of the licensor’s choosing.

The LGPL was created by The Free Software Foundation to provide for an alternate licensing arrangement for software libraries. As discussed above, under the terms of the GPL, if a person were to link a library licensed under the GPL to an application program to form an executable, the entire executable would, arguably, have to be distributed under the terms of the GPL.⁵ In recognition of the fact that some developers may wish to allow others to link non-GPL licensed code with their libraries, The Free Software Foundation created the LGPL v2.1.

⁵ The word “arguably” is used in recognition of the debate over whether the dynamic linking of GPL-licensed libraries with other code would trigger the GPL requirement that the entire resulting executable be licensed under the GPL. A complete analysis of this debate is beyond the scope of this paper, however, it centres on the absence of any operative language in GPL v2 regarding linking and by confusing use of the differing concepts of “derivative works” and “collective works”. In GPL v3, The Free Software Foundation does not use such terms and has included a specific reference to linked code within the definition of “Corresponding Source”. Yet, despite the attempts to clarify the treatment of linked libraries, it is not clear that all ambiguities have been resolved. For further discussion of this issue, see [GNU General Public License Version 3: A GNU Addition to the Free Software License Genus](#) by Eric Smith, a copy of which can be downloaded at www.fmc-law.com.

Under the terms of LGPL v2.1, “you may also combine or link a ‘work that uses the Library’ with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer’s own use and reverse engineering for debugging such modifications” (Section 6 of LGPL v.2.1). Read on their own, it is unclear whether these words require that the distributor must make the source to the “work which uses the Library” available so that the licensee can modify such work. However, if such interpretation were correct, it would mean that LGPLv2.1 would have essentially the same affect as GPLv2. The remainder of Section 6 of the LGPL makes clear that while the Library must be distributed under the terms of the LGPL, the distributor must only distribute the object code version of the “work that uses the Library”, and must do so in a manner that the licensee can modify the Library and then re-link it with the object code of the “work that uses the Library.”

(c) Non-FOSS Licenses

As important as it is to understand what constitutes FOSS, it is equally important to recognize that there are many examples of non-FOSS that are also readily available to software developers for use in their development activities. Some non-FOSS licenses, such as the Sun Community Source License and some of the licenses created by Microsoft under the Microsoft Shared Source Initiative⁶, provide access to source code for certain software, but do not allow the recipient to freely modify and/or redistribute such code.

Another example of non-FOSS is freeware. Not to be confused with free software, freeware is a term that is typically applied to software that is made available for use at no cost (i.e. free beer as opposed to free speech), though it may be accompanied by a request for a

⁶ Two of the licenses under the Microsoft Shared Source Initiative, the Microsoft Public License and the Microsoft Reciprocal License, have been approved by the Open Source Initiative as satisfying the Open Source definition.

donation. While some Freeware may also be FOSS, there are many examples of Freeware that do not include source code and are distributed pursuant to licenses that contain any number of restrictions on use and redistribution.

Shareware is another type of software or distribution model that is often confused with FOSS. The term “shareware” is customarily used to describe software that is distributed for use for a time-limited period (or trial period) at no cost to the recipient, and which often has limited functionality or restricted uses. After the trial period has expired, users must pay a license fee in order to continue to use the software and/or gain access to its full functionality. Again, source code is rarely provided as part of shareware.

Outside of the specific labels given to non-FOSS is a general category, sometimes referred to as “commercial” or “proprietary” software. While the labels are misleading (for example, FOSS is “proprietary” in the sense that there is one or more copyright holder in each item of FOSS), the terms have become known as referring to software distributed by for-profit corporations under a “closed-source” distribution model. In such cases, the suppliers typically distribute copies of their software for a fee and license its use under a restrictive license that limits the manner in which the software can be used, limits reproduction of the software to making an archival copy, if at all, and does not permit modification or redistribution of the software.

Why Emphasize Free and Open Source Software?

As the use of all externally developed software (i.e. software that is developed by others and the copyright for which has not been assigned to the user, by contract or operation of law) is subject to the exclusive rights of the holder(s) of the copyright in such software⁷, and any

⁷ Use of inventions embodied in software is also subject to the rights of the holders of patents for such inventions.

conditions that such copyright holders may place on licenses they grant to use such software, it not always clear why special attention is given to FOSS when discussing the need to manage the use and distribution of externally developed software. However, for many organizations, managing the use of FOSS is often more problematic than managing the use of non-FOSS acquired through more traditional channels.

When acquiring a license to non-FOSS, the individual within the organization that wants to acquire such a license typically has to obtain certain approvals from within the organization. If, as is typically the case, there is a fee associated with obtaining the license, he or she will often have to obtain the approval of the project leader who has oversight of the budget for the project. While low cost licenses may not require any negotiations, higher cost licenses will typically involve some bargaining that will involve other individuals within the organization, such as a business manager or financial officer. As part of these negotiations, the individuals involved in the negotiations will typically also have the terms and conditions of the applicable license reviewed by legal counsel. Though not all of these steps are necessarily followed when obtaining a license to non-FOSS, there is a greater chance that individuals within the organization, including the applicable software developers, will become aware of the intention to use the code and the restrictions and obligations that accompany the license to use the software.

In the case of FOSS, there is typically no financial cost associated with obtaining a license to use such code, and together with the availability of source code, this makes it easier for software developers to integrate such code into products without proper oversight. As a result, many organizations have discovered that, unbeknownst to all but one, or very few, individuals, their IT infrastructure and/or products contain FOSS.

Why Should You Care about Managing Use of FOSS?

While the use, modification and distribution of FOSS may have once been an issue ignored by most organizations, it is no longer. The use of FOSS, and compliance with FOSS licenses, is increasingly becoming an important issue for copyright holders, customers, potential acquirers and investors. Failure to develop and implement processes to manage the use, modification and use of FOSS can have serious consequences.

(a) Increase in Enforcement Actions

While the risk of facing a lawsuit as a result of one's non-compliance with the terms of a FOSS license was once considered remote, recent lawsuits have shown that the copyright holders of FOSS are willing to take action to enforce their copyright and the terms of their licenses. One of the principal players in these lawsuits is the Software Freedom Law Center ("SFLC").

Founded in 2005, the SFLC provides support to FOSS projects that are in need of legal, financial and administrative support. In September 2007, the SFLC filed the first ever U.S. copyright infringement lawsuit based on a violation of the GPL on behalf of two developers of "BusyBox", a program that provides Unix system utilities for use in embedded systems. The suit, *Erik Anderson and Rob Landley v. Monsoon Multimedia Inc.* claimed that Monsoon Multimedia, a distributor of wireless video streaming devices, infringed the copyright of the developers of BusyBox by distributing the BusyBox program in its devices and, in violation of GPL v.2, the license under which BusyBox is distributed, failing to make the source code available to recipients of the Monsoon device.

Approximately one month after the lawsuit was filed, the parties reached a settlement which resulted in the lawsuit being dismissed and the reinstatement of Monsoon Multimedia's license to distribute BusyBox. Monsoon Multimedia also agreed to appoint an Open Source

Compliance Officer within its company, publish the source code for the version of BusyBox it previously distributed on its Web site, and to notify previous recipients of Monsoon Multimedia device containing the BusyBox program of their rights with respect to the BusyBox program under the GPL. The settlement also provided for an undisclosed amount of financial consideration to be paid by Monsoon Multimedia to the plaintiffs.

Since the initial lawsuit related to the BusyBox program, the SFLC has filed numerous other lawsuits on behalf the developers of BusyBox. While some have not reached conclusion, most have been settled on terms similar to those in the Monsoon Multimedia settlement.

In addition to the BusyBox litigation, the SFLC filed a lawsuit against Cisco Systems, Inc. on behalf of The Free Software Foundation in December, 2008. In the suit, The Free Software Foundation alleges that Cisco distributed several of its GPL-licensed programs without providing complete and corresponding source code as required by the GPL and LGPL, including GNU C Library, GNU Coreutils, GNU Readline, GNU Parted, GNU Wget, GNU Compiler Collection, GNU Binutils, and GNU Debugger. In its request for relief, The Free Software Foundation has required an injunction as well as an award of damages and litigation costs. As of the date of writing, the litigation is on-going.

(b) Judgment Regarding Remedies

As they have rarely been tested in the courts, there has always been a degree of uncertainty regarding the enforceability and nature of FOSS licenses. Of paramount importance to FOSS licensors is whether they can obtain injunctions for copyright infringement against those who do not comply with the conditions of their FOSS license, or whether they are limited to claims for damages for breach of contract. As most FOSS is licensed without license fees or royalties, it is feared that the threat of a claim for damages for breach of contract would be a

weak deterrent to non-compliant licensees. While this issue remains a concern, the advocates of FOSS have received, in the form of the U.S. Court of Appeals decision in the case of *Robert Jacobson v. Matthew Katzer and Kamind Associates, Inc. (doing business as KAM Industries)*, some support for the position that an injunction is an appropriate remedy.

Jacobson's allegation that portions of his "DeCoder Pro" software, licensed under the terms of the Artistic License, were copied, modified, and distributed as part of the defendants' competing software known as, "Decoder Commander", was not in dispute. Also not in dispute was the fact that in copying, modifying and distributing the DeCoder Pro software, the defendants did not comply with the attribution and identification requirements set forth in the Artistic License.

Jacobson brought an action before the United States District Court for the North District of California for copyright infringement and moved for a preliminary injunction. In denying the motion for a preliminary injunction, the District Court held that the Artistic License was a license intentional broad and unlimited and:

"[t]he condition that the user insert a prominent notice of attribution does not limit the scope of the license. Rather, Defendants' alleged violation of the conditions of the license may have constituted a breach of the nonexclusive license, but does not create liability for copyright infringement where it would not otherwise exist."

In the opinion of the District Court, the conditions set out in the license were merely covenants of the licensee that fell outside of, and did not affect, the scope of the license. Defendants' breach of such covenants gave rise to a claim of breach of contract, and such a breach does not give rise to a presumption of irreparable harm, one of the required elements for a preliminary injunction to be granted.

In vacating the decision of the District Court, the U.S. Court of Appeals concluded that the conditions of the Artistic License are to be regarded as restrictions on the scope of the license and not merely covenants of the licensee. In explaining its decision, the Court stated that “[c]opyright holders who engage in open source licensing have the right to control the modification and distribution of copyrighted materials....Copyright licenses are designed to support the right to exclude; money damages alone do not support or enforce that right.” The Court further stated, “because a calculation of damages is inherently speculative, these types of license restrictions might well be rendered meaningless absent the ability to enforce through injunctive relief.

While the *Katzer* decision dealt only with the Artistic License, it is not hard to see the same analysis being applied to similar licenses, such as the GPL. While this decision is likely not the last word on remedies available for non-compliance with FOSS licenses, it does serve as a warning to users of FOSS that non-compliance can result in the award of an injunction against its further distribution of the affected FOSS, and any product in which it is incorporated.

(c) People Are Asking about FOSS Use

Despite the recent increase in FOSS related litigation, the number of cases remains relatively small. For some organizations, the risk of litigation may still seem remote, and the time and cost of establishing and implementing a FOSS management process may cause many organizations to delay such actions. However, even if FOSS management is not a priority for an organization, it is increasingly becoming a priority for those with whom organizations deal. As a result, FOSS management is becoming less a matter of choice, and more a matter of necessity.

(i) Customers

One group that is increasingly asking questions about the use of FOSS is customers/licensees. More precisely, they want to know if the products being sold or licensed to them include FOSS, whether the supplier has complied with the terms of the applicable FOSS licenses and, if the customer will be redistributing such product, with or without modification, whether such licenses will affect their redistribution of such product and/or of any larger works in which such software is incorporated.

In addition to concerns regarding the FOSS license(s) under which the FOSS was obtained, customers are also concerned with issues such as: (i) the origins of the FOSS, the reliability of the software programmers and the continued viability of the FOSS project; (ii) potential intellectual property infringement, including patent infringement; (iii) lack of indemnification for infringement claims (unless the supplier will provide such protection); (iv) the shifting of risk of errors, viruses, etc. to the licensee (i.e. no warranties), unless provided from the supplier; (v) the "reciprocal" or "viral" nature of some FOSS licenses (e.g. GPL); (vi) inadvertent granting of patent rights by distributing FOSS; (vii) patent retaliation provisions in some FOSS licenses; and (viii) incompatibility of some FOSS licenses with other licenses, including other FOSS licenses.

Due to these concerns, customers are increasingly asking suppliers to disclose the use of FOSS in their products. This disclosure may come in the form of warranties that are included in the purchasing/licensing agreements between the customer and supplier, and/or in questionnaires to be completed by the supplier as part of the procurement process. Some organizations also require suppliers to fill out a certificate of compliance for each version of product supplied to them, which certificate provides details regarding FOSS that is included in the product(s), the

licenses under which the FOSS was obtained by the supplier, as well as warranty that the supplier has complied with the terms of the applicable FOSS licenses. Finally, an increasing number of customers are requiring suppliers to provide to them the results of audits of the product source code using automated scanning tools that are now available, or requesting that they, or an independent third party, be permitted to perform the audit.

(ii) Acquirers and Investors

In addition to customers, potential acquirers and investors are also concerned with an organizations use of FOSS. Potential acquirers do not want to acquire a potential lawsuit or other risk item as a resulting of unknowingly acquiring a company or technology that has problems arising from its use of FOSS. As a result, FOSS is increasingly become the subject of pre-acquisition due diligence requests. As in the case of customers mentioned above, the due diligence process increasingly includes the auditing of source code using automated tools which assist in the detection of FOSS.

Similar to potential acquirers, those financing technology-based businesses do not want to finance a business that may have serious issues arising from its use and distribution of FOSS. As a result, sophisticated investors are asking for the same types of warranties and disclosures regarding FOSS as are customers and potential acquirers.

The increasing awareness on the part of acquirers and investors regarding FOSS use means that a lack of FOSS management processes can be costly to an organization. The discovery of an FOSS issue during the due diligence process, such as the failure to comply with the terms of a FOSS license, can dramatically affect the valuation that has been attributed to the organization or its assets. Such a discovery can also affect the timing of the transaction, as the acquirer or investor may insist that the issue be resolved prior to closing. For issues that cannot

be resolved prior to closing, or when closing simply cannot wait, the target company and/or its shareholders may be required to indemnify the purchaser or investors for some or all losses that might arise from the unresolved issue. In addition, a portion of the consideration may be withheld as security for such indemnities. Finally, in the worst cases, a prospective purchaser or investor may refuse to proceed with the transaction as a result of FOSS issues discovered during due diligence.

Managing the Procurement, Use and Distribution of FOSS

For the reasons outlined above, the need to manage the procurement, use and distribution of FOSS is clear. Unfortunately, when talking about this issue with clients, many initially think that all they need to do to is develop an FOSS policy. While an FOSS policy is an important component of managing the procurement, use and distribution of FOSS, it does not represent the entire process.

While each organization will need to tailor its management processes to meet its own needs and structures, some of the common elements found in most FOSS management processes included the following:

(a) FOSS Committee

A committee to oversee the development and implementation of FOSS management processes should be created. As the management process will involve multiple departments within the organization, the committee should include representation from each affected department. This would typically include representation from software development, product management, legal, auditing and risk management, and perhaps others. As committee members will typically have other responsibilities as part of their jobs, it is recommended that

organizations, resources permitting, appoint an individual whose sole responsibility is the FOSS management process. This individual should have ownership over all aspects of the process and should report to a senior officer of the company, or the board of directors, on a regular basis regarding the development and implementation of the FOSS management policies and procedures.

(b) FOSS Policy

The development of an FOSS policy is a key part of any FOSS management process. The policy should educate personnel regarding the benefits and risks of FOSS to the organization and clearly set the organization's expectations of its personnel when it comes to using FOSS. The policy should also set out the process for approving the procurement and use of FOSS, as well as the procedures to be used by the organization to ensure compliance with the policy. The policy should also address what procedures will be used to ensure that outsourced development work, or proposed acquisition targets meet the criteria set out in the policy.

The organization may consider publishing the policy in draft form and soliciting comment from its personnel before finalizing. Once finalized, the policy should be published within the organization and used as a basis for educating personnel regarding the organizations positions on using FOSS and each individual's responsibilities with respect to FOSS. The consequences of failure to adhere to the organization's FOSS policies and procedures should also be made clear.

(c) Approval Process

The approval process should be structured in manner that allows for an efficient assessment of each request to use FOSS and that results in a well-reasoned "Yes" or "No" response. Again, the approval process should be tailored to the structure and objectives of the

organization, but in creating an approval process, the following elements should be considered:

- (i) Review of the supplier (e.g. Red Hat v.s. "guy in his basement") – elements such as the size and age of the FOSS project should be considered. In addition, whether the FOSS in question is widely used by other organizations can be an important factor;
- (ii) Search for known issues (e.g. security or export control issues, any known, or alleged, intellectual property infringement);
- (iii) Quality control standards - does organization have minimum requirements for code quality, documentation, support, warranties or indemnities?;
- (iv) Identify intended use of FOSS (e.g. internal use only, modification, combination with other code, redistribution);
- (v) Identify any other sources of code which address need, including potential for developing in-house or outsourcing and assess the risks and benefits of each alternative; and
- (vi) Review of license terms for FOSS - is intended use permissible? Are there license incompatibility issues? Is the license of the "viral" variety and how will the intended use affect other code developed or used by the organization? Does the license include mandatory patent license or patent retaliation provisions? Identify compliance obligations, such as notice requirements and making the source code to the FOSS available to down-stream licensees of the FOSS.

(d) Guidelines for Making Modifications

The FOSS policy should also include guidelines for making and approving modifications to FOSS. Questions to be considered include: will the organization be making modifications to the FOSS? If so, are modifications for internal use only or for redistribution? If required by the license terms, is the organization comfortable with releasing the source code to such modifications?

(e) Contributions to the FOSS Community

The FOSS policy should prohibit contributions of the organization's source code to open source projects without approval. If the organization wishes to contribute code to a FOSS project, or to release "proprietary" code under a FOSS license, a full analysis needs to be conducted, including ensuring that the organization has all necessary rights to disclose the source code to others and that the appropriate FOSS license terms are chosen. The FOSS policy should also set out rules and procedures for employees who wish to participate in an FOSS project that is unrelated to his or her job, including requiring prior disclosure of such participation, that he or she must not use the organization's resources and equipment in doing so, and that he or she must not identify his employer without the employer's consent.

(f) Licensing and M&A Transactions

An organization's FOSS policy should set out the standard of due diligence and other procedures it requires when licensing or acquiring technology from third parties, or considering an investment or acquisition of a company. At a minimum, the applicable transaction documents should include appropriate representations and warranties regarding the use of FOSS software (with appropriate carve-outs from any caps on liability). It is also useful to have the licensor or target company complete a short FOSS questionnaire as part of the due diligence process so as to

highlight the importance of the issue and focus attention not only on the use of FOSS, but also to try to gain an understanding of the manner in which it is being used and under what license terms. An automated audit of the applicable source code is a fast and efficient way to verify the answers that are received from the licensor or target company.

(g) Compliance with FOSS Policy

A policy is only useful if it is diligently followed. As a result, it is important that the organization implement measures that allow it to track its adherence to its FOSS policy. One common method for tracking compliance is the performance of regular audits of software development projects (e.g. using automated auditing tools) for the presence of FOSS and comparing the results to the approvals granted during the approval process. In addition to verifying that all components found in the audited software have been approved, it is equally important to verify that the organization has complied with all of its obligations regarding the distribution of FOSS included in the audited software. If non-compliance is identified, the organization must develop a plan for remedying such non-compliance.

In addition to the obvious distribution obligations, such as notice requirements and making source code available (as applicable), an often overlooked aspect of distributing FOSS as part of a larger product is ensuring that the grant of license in the distributing company's software license is correct. For example, non-reciprocal FOSS licenses such as the BSD license state that redistribution and use of the code, with or without modification, is permitted provided certain conditions are met. One such condition is that the "copyright notice, this list of conditions and the following disclaimer" must be retained in all copies of the FOSS that are distributed. What is unclear is if the retention of such items means that "down-stream" recipients of the code are granted a license to redistribute, modify and use the code from the copyright

holder, or whether the license is granted by the redistributing entity by way of sublicense.

The answer to the above question will impact the manner in which redistributors of affected FOSS draft their own licenses. If the redistributor is merely passing along the license from the copyright holder, it would be improper for the redistributor to state that it is granting to a third party a license to use the entire work. Instead, it would have to include a carve-out for the affected FOSS and state that the license to use, modify and redistribute the affected FOSS is granted by the copyright holder to the licensee. Carefully drafting will have to be practiced by the parties so that such a carve-out will not have unintended consequences (e.g. defining a term such as "Licensor's Software" to exclude the FOSS, with the result that warranties, support or indemnification obligations do not apply to the FOSS).

Summary

Due to the increase in awareness of issues that can arise from using FOSS, the impact of its use on your organization can no longer be ignored. Remaining in the dark regarding the presence of FOSS in your organization IT infrastructure and products can result in disastrous consequences. Only through the development and implementation of sound policies and procedures regarding the procurement, use, modification and distribution of FOSS can these risks be properly managed.