International
Open
Source
Network

ELSEVIER

UNDP

Free/Open Source Software

Licensing

Shun-ling Chen

Asia-Pacific Development Information Programme
e-Primers on Free/Open Source Software

Free/Open Source Software Licensing

Shun-ling Chen

Asia-Pacific Development Information Programme
e-Primers on Free/Open Source Software

ELSEVIER
A division of
Reed Elsevier India Private Limited
Shri Pratap Udyog, 274 Capt Gaur Marg,
Sriniwas Puri, New Delhi 110 065
Tel: +91 11 42890200
Fax: +91 11 42890201
Website: www.asiaelsevier.com

ISBN-13: 978-81-312-0422-1
ISBN-10: 81-312-0422-7

Academic Press, Butterworth-Heinemann, Digital Press, Focal Press, Morgan Kaufmann, North Holland and Pergamon are the Science and Technology imprints of Elsevier.

Printed and bound in India.

IOSN is an initiative of UNDP-APDIP; its statements, documents and initiatives do not necessarily reflect the official viewpoint or policies of UNDP.

TABLE OF CONTENT

PREFACE

Under the current copyright regime, software users are subject to restrictive regulations. This regime is said to provide individuals with incentives in terms of economic returns and thus encourages them to develop creative works. However, some software developers disagree with this default configuration of copyright law and value other things more than short-term economic incentives.

The Free/Open Source Software (FOSS) Movement was started by grassroots developers who are not content with the current copyright system. They tactically use specifically designed FOSS licenses to allow a community with a different world-view to develop and flourish. Lawyers are sometimes brought in to facilitate the collaboration between developers.

However, software development and software licensing are very different activities, and developers and lawyers often have very different mindsets. While developers tend to use whatever resources are available to them to achieve a particular feature, lawyers may request a copy of the license of every existing module that developers wish to adopt, before they actually approve the integration into the project. And while developers tend to use acronyms to make their communication more succinct, lawyers tend to use arcane terms and complicated sentences to make sure their ideas can be clearly delivered. Therefore, in order to successfully develop FOSS applications, both these professions are required to cooperate with each other.

As the FOSS Movement has been growing rapidly in recent years, more and more different kinds of stakeholders are brought in to participate in different roles. Some of them are end-users, developers, business entities, or government agencies that provide funding for FOSS projects. This primer is designed to provide these stakeholders with some basic knowledge about copyright, software copyright and FOSS licenses. Legal issues may vary in different situations and this primer may not be able to provide answers to all situations. But, hopefully, it will serve as a bridge between lawyers and non-lawyers in this joint venture of FOSS development.

INTRODUCTION

A general introduction to free/open source software (FOSS) is provided in the first of this series of primers. That first primer is available at the International Open Source Network (IOSN) website at:

http://www.iosn.net/foss/foss-general-primer/foss_primer_current.pdf

Under the prevailing copyright regime, licenses decide whether software can be free and/or open. As David Wheeler said, FOSS are programs whose licenses give users the freedom to run them for any purpose, to study and modify them, and to redistribute copies of either the original or modified programs without having to pay royalties to original developers.[1]

Starting from the mid-1980s, the birth of the GNU General Public License (GNU GPL or GPL) has enabled a model for software development.[2] Following GNU GPL, various FOSS licenses have been drafted and adopted by FOSS communities, academic institutes and commercial companies. The number of FOSS licenses is growing rapidly. In early 2003, 43 licenses were recognized by the Open Source Initiative (OSI) as open source licenses. A year-and-a-half later, in July 2004, the number had reached 54. The diversity among FOSS licenses sometimes causes confusion and difficulty for people who want to participate in FOSS projects or adopt FOSS solutions. Some have argued that to reduce the transaction cost, new licenses should not be created. However, the number of FOSS licenses has been growing.[3]

This primer aims to provide an introduction to FOSS licensing issues. It begins with a brief overview of "intellectual property" rights,[4] and then moves on to the development of copyright law, the category of "intellectual property" that is most relevant here. The primer will then examine different proprietary and FOSS licenses which use copyright law to regulate the use of software. Finally, it briefly explains how the FOSS movement uses licenses as a way to create a different model of software development.

There are many FOSS licenses, and they may differ from each other in major ways. Due to page limitations, however, only three pervasively adopted licenses are discussed in this primer: the GNU GPL, the GNU Lesser General Public License (LGPL) and the Berkeley Software Distribution (BSD) style licenses. These three are important not only because a large number of FOSS projects are under licenses, but also because they represent very different styles of FOSS licensing.

In the last section of the primer, some scenarios are given to highlight possible copyright issues regarding the use of FOSS by end-users, developers and vendors. Given the increased attention paid by governments to FOSS development, the primer also includes two cases regarding government-sponsored FOSS projects.

[1] Wheeler, D., "Why OSS/FS? Look at the Numbers!" Available from http://www.dwheeler.com/oss_fs_why.html; accessed on 7 November 2003.

[2] GNU is a recursive acronym for "Gnu's not Unix".

[3] For example, five licenses were approved in February 2004, and two licenses were added to the list of approved licenses in June 2004. Available from http://www.opensource.org/weblog/2004/01/03#newsblog and opensource.org/weblog/2004/06/03#Jun2-04; accessed on 5 July 2004.

[4] The term "Intellectual Property" covers different areas of law such as Copyright, Patent, and Trademark. Some people, especially free software advocates, advise against using the term because they believe that these different areas of law cannot be generalized. Another reason to object to its usage is that the term implies that these disparate legal issues are taken as based on an analogy of the property rights to tangible objects, whereas software is intangible. See, for example, "Some Confusing or Loaded Words and Phrases that are Worth Avoiding." Available from http://www.fsf.org/philosophy/words-to-avoid.html; accessed on 9 November 2004.
It is true that the term "intellectual property" is relatively new and is loaded with the above meaning. Nevertheless, since the existing legal structure does take intangible objects as tangible objects, the term is still used in this primer but is within quotation marks to draw attention to these critical opinions.

AN OVERVIEW OF “INTELLECTUAL PROPERTY RIGHTS”

Intangible products of human creative activities are regarded as a kind of property and are granted protection in the same way as property rights have been traditionally protected and applied to tangible objects.

Copyright, patent, trademark and trade secret all fall under the category of “intellectual property”. But each must be understood to be significantly distinct from the others.

- **Trade Secret**

  A trade secret is a confidential practice, method, process, design, the “know-how” or other information used by a business to compete with other businesses. The precise language by which a trade secret is defined varies by jurisdiction. However, there are three factors that (though subject to different interpretations) are common to all such definitions: a trade secret is some sort of information that is not generally known to the relevant portion of the public; confers some sort of economic benefit on its holder; is the subject of reasonable efforts to maintain its secrecy. Trade secrets are regulated by using a variety of civil and commercial means, such as confidentiality or non-disclosure agreements signed by those who are given access to special knowledge and information.[5]

- **Trademark**

  Trademarks are the distinctive names, phrases, symbols, designs, pictures or styles used by a business to identify itself and its products or services to its consumers. In many countries, colours, three-dimensional marks, sounds, and even smells can also be trademarked.[6]

- **Patent**

  While trade secrets enable a business to keep certain information from the public, patents are designed to grant the inventor monopoly rights or monopoly status over certain newly developed knowledge for a period of time (usually 20 years) in exchange for its disclosure. Typically, to gain such rights, the inventor is required to file a patent application, which will be reviewed by a designated patent examiner. Novelty of the invention is an essential criterion in granting a patent.[7]

- **Copyright**

  Copyright is applied to various kinds of creative works, such as literary works, music compositions, paintings and software. Unlike patents, copyright applies to a work upon its creation, regardless of its novelty.

  However, the ideas employed by the work cannot be copyrighted. Copyright only prevents others from copying the copyright holder’s particular way of expressing those ideas.[8] Under Copyright Law, the copyright holder is entitled to exclusive rights of reproduction, modification, distribution, and public display and performance of her copyrighted work. A license is often used to explain under which terms and conditions the work can be used. To accommodate different situations, the copyright holder is entitled to draft and adopt different kinds of licenses for each piece of her work.

[5] Available from http://en.wikipedia.org/wiki/Trade_secret; accessed on 27 July 2004.

[6] Available from http://en.wikipedia.org/wiki/Trademark; accessed on 27 July 2004.

[7] Available from http://en.wikipedia.org/wiki/Patents; accessed on 27 July 2004.

[8] Available from http://en.wikipedia.org/wiki/Copyright; accessed on 27 July 2004.

How is software regulated?

Software is now subject to Copyright Law. Moreover, in recent years it has been argued that software should be patentable as well. Although software patents have been granted in some cases, they are still questioned by many, especially by the FOSS community. Due to page limits and the complexity of the issue, this primer does not address this topic.

COPYRIGHT BASICS

Why do We have Copyright?

Compared to other legal concepts, copyright is a relatively new invention in human history. The development of copyright regulation reflects the social and technological transformation around human creative activity and distribution of the resultant profits. While granting exclusive private rights to authors or copyright holders has been considered as a way of encouraging human creative activity, copyright law also claims to recognize the larger public interest, particularly with respect to education, research and access to information.[9]

Copyright law uses various means to balance public and private interests. In the Statute of Anne (1710), the earliest modern copyright law, authorities are allowed to limit and control the price of printed books according to their best judgement. In the United States Constitution, authors are granted exclusive rights to their writings within a limited time. In copyright law, fair use exceptions are specified to avoid the drawbacks of excessive assertion of exclusive rights and to attain a balance between conflicting interests.

What can be Copyrighted?

Copyright applies to the expression of ideas in different forms, including literary, dramatic, musical, artistic, and other intellectual works.[10] The ideas expressed in such works are themselves not copyrightable.[11] Since the 1980s, the copyrightability of software became internationally accepted.[12]

How do I Copyright my Work?

Nowadays, copyright law does not require formalities. The author does not need to publish, register, pay a registration fee of any kind, nor attach a copyright notice to her work, for her copyright to take effect. Copyright is automatically applied to a work once it is created[13] and the creator of the work automatically becomes the copyright holder.

What Rights are Granted to the Copyright Holder?

Copyright is actually a bundle of rights, including the right to reproduce the work, to prepare derivative works based on it, to distribute copies of a work, and to perform and display a copyrighted work publicly. A few other kinds of rights are defined in copyright law.[14] Without the consent of the copyright holder, it is illegal for anyone to perform any of the activities mentioned above.

The Expansion of Copyright Law

Copyright Law has been expanded with time.

The first copyright legislation (the Statute of Anne, 1710): Compared to other legal systems, copyright law came relatively late in human civilization. The first known copyright legislation was the Statute of Anne, enacted in 1710 in Great Britain.[15] For a newly created work, the Statute of Anne granted the

[9] As stated in the Preamble of WIPO Copyright Treaty. Available from http://www.wipo.int/clea/docs/en/wo/wo033en.htm; accessed on 29 June 2004.

[10] Available from http://www.wipo.int/copyright/en/faq/faqs.htm#rights; accessed on 28 June 2004.

[11] Available from http://www.wipo.int/copyright/en/faq/faqs.htm#ideas; accessed on 28 June 2004.

[12] Available from http://www.wipo.int/copyright/en/faq/faqs.htm#P39_5114; accessed on 28 June 2004.[1]

[13] Different jurisdiction may have been set at different points when copyright came into existence. Some jurisdiction may require the work to be fixed, others may only ask the work to be finished.

[14] Available from http://www.wipo.int/copyright/en/faq/faqs.htm#rights; accessed on 28 June 2004.

[15] Available from http://www.copyrighthistory.com/anne.html; accessed on 28 June 2004.

copyright holders the right to print and reprint books and other writings for 14 years.

All-dimensional expansion of copyright law: We can see from the Statute of Anne that, initially, the scope of copyright was quite limited. The copyrightable works were limited to books and other writings, the rights granted to the copyright holder were limited to printing and reprinting the work, and the length of the protection was limited to 14 years. Now, copyright law does much more. Copyrightable works now include paintings, sculpture, music compositions, music recording, architecture, and software. The bundle of rights granted to the copyright holder have been expanded to include the right to print, reprint, modify, display publicly, perform publicly, and distribute the work. Moreover, the term of copyright protection has been increased to 50 years after the author's death. (In Europe and the US, it has been expanded to 70 years.)[16]

From National to International

The expansion of copyright law has not been limited to one jurisdiction; it has become standardized internationally.

Berne Convention: In the late 19th century, as copyrighted works gradually became important in international trade, the transnational copyright system gradually became a serious issue. The Berne Convention of 1886 first introduced the national treatment principle. This means that signatories to the Berne Convention will treat the work of a foreign copyright holder just as they treat their own citizens' work. Thus, it created an international standard for copyright regulation.

However, without a dispute resolution mechanism, the Berne Convention offered a somewhat weak copyright, as it will be too costly for copyright holders to claim their rights in a foreign country where they believed their rights had been infringed.

More enforceable international standard: WTO and TRIPs: In the 1990s, the World Trade Organization (WTO) and the Trade Related Intellectual Property Agreement (TRIPs) sought to establish a stronger international copyright regime. Every economy intending to become a WTO member is required to sign the TRIPs, and every TRIPs signatory must agree to comply with all of the key sections of the Berne Convention. The WTO also provides a dispute-settlement and enforcement mechanism for copyright infringements among member countries. Thus, copyright has become more enforceable internationally.[17]

The Abolition of Formality Requirements

As set forth in the 1908 Berne Convention, copyright is applied to a work once it is created, without the need for any formality.[18] The author is not required to register, or even to publish a work to enjoy full copyright protection. In Berne Convention signatory countries, the law assumes that all authors claim all rights granted to them unless they explicitly state otherwise.

Copyright laws in different countries have been revised to comply with this standard. For example, anticipating that it would join the Berne Convention Union, the US revised its Copyright Act and abolished formality requirements in 1976.[19]

[16] Little, J., "History of Copyright- A Chronology," 2002; available from http://www.musicjournal.org/01copyright.html; accessed on 28 June 2004.

[17] Story, A., "Don't Ignore Copyright, the 'Sleeping Giant' on the TRIPs and International Educational Agenda," pp.132-33, in Drahos, P. and Mayne, R. (eds.), *Global Intellectual Property Rights, Knowledge, Access and Development*, NY: MacMillan, 2002.

[18] Lessig, L., "Free Culture," Footnote 194; available from http://www.jus.uio.no/sisu/freeculture.lawrence.lessig/14; accessed on 29 June 2004.

[19] Little, J., "History of Copyright- A Chronology", 2002; available from http://www.musicjournal.org/01copyright.html; accessed on 28 June 2004.

SOFTWARE AND COPYRIGHT

Extension of Copyright Law to Software

Since 1980, it has become an international trend for copyright to be applied to computer software. The WIPO Copyright Treaty (1996) also states that computer software should be regulated by copyright law.

Copyrightability of the Source Code and the Object Code

Software can be expressed in both source code and object code. However, TRIPs states that software copyright applies to both forms.[20] In practice, proprietary software companies tend to release their product only in object code, and keep the source code of the product as their trade secret.[21]

As mentioned earlier, under copyright law only the expression of ideas but not the ideas themselves can be copyrighted. For example, with a literary work or a music composition, although the form of the work is copyrighted, other people can use the ideas expressed in the work as inspiration for their new works. With software, ideas can only be perceived by reading the source code. Since the source code is not often accessible, in effect the proprietary company is able to withhold the ideas that underpin the software. This contravenes the principle that only form and not the idea should be exclusively owned – a principle designed to maintain the balance between private and public interests.

Users' Rights Denied in Proprietary Licensing Models

The revision of the US copyright law in 1976 and shifts in the information technology (IT) industry in the 1970s changed the practices of software distribution. Before that, users received both source and binary object codes. Today, under proprietary license models, proprietary software companies usually make the source code inaccessible to ensure and maximize profit. Studying a proprietary software is usually explicitly prohibited. For example, even in licenses for developers, for example, Microsoft End User Agreement and Microsoft Developer Network Subscription, reverse-engineering, decompilation and disassembly are not allowed except and only to the extent that it is expressly permitted by the applicable law.[22]

For end-users, proprietary licenses usually allow only one copy of the software for each computer. That means, if the user has one desktop and one laptop, or two desktops, she will have to purchase two copies if she wants to run the program legally on both machines. If there are defects in the program that she has legally purchased, her only recourse is to contact the proprietary company regarding these defects. She will not be able to legally debug the program herself, or use unofficial patches, since modification of the program is not allowed. In effect, users of proprietary software are completely dependent upon the vendor.

Under the traditional proprietary licensing model, end-users were not able to protect their interest in a cooperative manner. The FOSS movement has contributed to the positive transformation of this situation. The Free Software Foundation (FSF), which was founded in 1985, is dedicated to promoting users' rights to use, study, copy, modify and/or redistribute computer programs.[23] These are the rights that are not usually granted to end-users in the licenses of proprietary software.

[20] Available from http://www.wto.org/english/docs_e/legal_e/legal_e.htm#TRIPs; accessed on 28 June 2004.

[21] Halligan, R. M., "How to Protect Intellectual Property Right in Computer Software;" available from http://my.execpc.com/~mhallign/computer.html; accessed on 1 July 2004.

[22] Available from http://www.msdnaa.net/EULA/NA/English.aspx; accessed on 4 August 2004.

[23] Available from http://www.fsf.org/fsf/fsf.htm; accessed on 4 August 2004.

HOW IS FOSS DIFFERENT FROM PROPRIETARY SOFTWARE?

The development of FOSS may be considered a reaction of the community of software developers to existing legal definitions of software copyright. For both free software and open source developers, access to the source code is a prerequisite to exercise the rights bundled in copyright, such as the right to make copies of a work, to distribute these copies, and to prepare derivative works.

Transition in IT Industry and Legal Institutions

In the 1970s, developments in legal institutions and the IT industry stimulated the formation of the free software movement in the US. For one thing, the US copyright law went through a major revision in 1976, and the question of whether software is copyrightable was put on the table under relentless pressure from IT companies.[24] Second, while software used to be bundled with hardware in the hardware market, the IT industry began to consider the software itself as a separate product.[25] At this point, IT companies began to recruit more developers from research institutes to develop software, and the companies asked these individuals to sign confidentiality agreements upon recruitment.

Richard Stallman on a Stark Moral Decision

Prior to the above transition, the common practice in laboratories was to share sources and copies. For Richard Stallman (RMS) who worked at the Massachusetts Institute of Technology (MIT) laboratory at the time, the change undermined the community that honoured sharing and the ethic of "helping your neighbours". For Stallman, a talented programmer who could easily sign a contract and a confidentiality agreement with a proprietary company in exchange for a well-paid salary, the "stark moral decision" was between private gain for himself (and proprietary software companies) or the survival and sustainability of the community of software developers. He chose the latter and began the Free Software movement.[26]

Free Software Defined

Free software is about granting users the freedom to run, copy, distribute, study, change and improve the software. Free software is any software that provided the following freedoms. The freedom to:

- Run the program, for any purpose (freedom 0).
- Study how the program works, and adapt it to your needs (freedom 1). Access to the source code is a precondition for this.
- Redistribute copies so you can help your neighbour (freedom 2).
- Improve the program, and release your improvements to the public, so that the whole community benefits (freedom 3). Access to the source code is a precondition for this.[27]

Apart from emphasizing access to the source code, the Free Software Definition also stipulates the user's right to copy, to distribute the copy, to modify the software, and to distribute the derivative work of a copyrighted work. All these rights are granted exclusively to the copyright holder under copyright law.

[24] Richard, J., "Copyright Protection for Computer Software in the United States," 2002; available from http://www.ladas.com/Patents/Computer/SoftwareAndCopyright/Softwa04.html; accessed on 28 June 2004.

[25] Campbell-Kelly, M., "Development and Structure of the International Software Industry, 1950-1990;" available from http://www.dcs.warwick.ac.uk/~mck/Personal/SoftIndy.pdf; accessed on 1 July 2004.

[26] Stallman, R., 1999, "The GNU Operating System and the Free Software Movement," pp.53-56, O'Reilly & Associates, Inc., Canada.

[27] Available from http://www.fsf.org/philosophy/free-sw.html; accessed on 31 May 2003.

Creating a Free Software Environment

The GNU Project and the Free Software Foundation

It is not sufficient to stipulate the rights of users or non-copyright holders. It is also important to have a computing environment in which these rights can be exercised. Thus, the GNU project was launched in 1984 to develop the GNU system, a complete UNIX-style free operating system. Today, the GNU project also includes other software applications.

In 1985, the Free Software Foundation (FSF) was established to promote the idea of free software. It promotes the development and use of free software not only by distributing free software, but also by encouraging the creation of a coherent system, the GNU operating system, and providing alternative solutions to proprietary software. For more information, see www.gnu.org/gnu/thegnuproject.html.

GNU General Public License

Under existing legal norms, once a work is created, copyright protection is granted exclusively to the copyright holder. Without an explicit expression, it is assumed that the copyright holder claims all the rights granted to her. The law burdens the copyright holder with explicit expression if she wishes to relinquish some or all rights granted to her.

Some people may not want to exercise all of the rights granted to them. However, they may not know how to make such an explicit expression. The GNU General Public License (GNU GPL) serves as a legal tool to help people to do so.

GNU GPL is a license. Unlike proprietary licenses, it grants users the rights that the law grants exclusively to the copyright holder. These include the right to access the source code; to run the program; to make copies and redistribute the copies; and to modify the program and distribute the modified program.

On the other hand, although GNU GPL grants the user many rights and freedoms to use the software, it also sets certain limitations on those who want to distribute the program or make and distribute derivative works to ensure that the software and its derivations will remain free.[28]

When a work is licensed under GNU GPL, it means that its author still claims copyright but adopts a different license as an explicit expression to allow the public to have greater freedom to use her work than what the copyright law allows by default.

Open Source Software

While free software advocates consider the four freedoms to be a moral issue, promoters of open source software focus more on the technical values and are, consequently, more business-friendly.[29] The Open Source Initiative (OSI) operates as an organization to promote the open source movement by managing and promoting the Open Source Definition (OSD) and its certification mark for open source licenses and products.

Open Source Definition

The OSD is a revision of a policy document of the Debian GNU/Linux Distribution that served to clarify which licenses are free licenses.[30] The OSI explains the basic idea of Open Source as:

> The basic idea behind open source is very simple: When programmers can read, redistribute, and modify the source code for a piece of software, the software evolves. People improve it, people adapt it, people fix bugs.[31]

The OSD echoes the rights stated in the Free Software Definition, including the users' access to the source

[28] See the Preamble of GNU GPL. Available from http://www.fsf.org/licenses/gpl.txt; accessed on 31 May 2004.

[29] Wong, K. and Sayo, P., *Free/Open Source Software, A General Introduction,* pp. 6-7, 2004; available from http://www.iosn.net/foss/foss-general-primer/foss_primer_current.pdf; accessed on 31 May 2004.

[30] Perens, B., "The Open Source Definition," in Open *Sources, Voice From the Open Source Revolution*, CA: O'Reilly & Associates, Inc., 1999.

[31] Available from http://www.opensource.org/; accessed 31 May 2004.

code (Section 2), the rights of users to copy the work and distribute the copies (Section 1), and the right to modify the work and distribute derivative works (Section 3).

The OSD also has several non-discrimination clauses (Sections 5, 6, 8, 9 and 10). Though not stated in the same way, these non-discriminatory ideas are also found in the Free Software Definition. Section 7 of the OSD aims to prevent the source code from being withheld by indirect means such as by requiring non-disclosure agreements. However, the emphasis on the integrity of the author's source code and the requirements for the distribution of modified works (Section 4) are not explicitly stated in the Free Software Definition. For details, see www.opensource.org/docs/definition.php.

OSI-approved Licenses

The OSI certifies and recognizes licenses as open source by following certain procedures. The certification is made upon request, and newly approved open source licenses are added to a list of open source licenses maintained by the OSI at www.opensource.org/licenses.

The number of OSI-approved licenses has been growing with the recent FOSS development. Some licenses are derived from the FOSS community: the GNU GPL, the LGPL, the PHP License and the Nethack General Public License. Those from academic/research institutes include the NASA Open Source Agreement, the MIT License and the University of Illinois/NCSA Open Source License. Some proprietary companies that have adopted FOSS as part of their strategies have also developed FOSS licenses including the Apple Public License, the Eclipse Public License, the Qt Public License and the Mozilla Public License. Actually, a large proportion of OSI-approved licenses are developed by for-profit companies.

Free or Restrictive?

Although the Free Software Definition and the Open Source Definition have much in common, they do differ in rhetoric, which reflects their differences in philosophy.

For example, some people may describe the GNU GPL and the LGPL as "highly restrictive" because the FSF set many restrictions to make sure that free software and their derivative works stay free. However, for the FSF, these restrictions are prerequisites for a healthy environment for free software.

The FSF also maintains a list of free software licenses and non-free software licenses. Although the FSF may sometimes describe these relatively simple licenses as "permissive", it never qualifies their more complicated ones as "restrictive".

Though there are philosophical differences, in most cases, the FSF and the OSI agree on the classification of FOSS and non-FOSS licenses. Twenty-six OSI-approved licenses have been analyzed by the FSF, and only two of these, the Original Artistic License and the Reciprocal Public License, are regarded as non-free licenses.

HOW TO MAKE THE SOURCE FREE/OPEN

Under current legal norms, software is protected by copyright law. Therefore, the FOSS movement has developed many different FOSS licenses to enable software developers to easily state that they grant their users some rights that copyright law grants exclusively to them. FOSS licenses also serve as agreements among FOSS developer communities.

There are many FOSS licenses, and their characteristics differ. In a later section of this primer, we will focus on three major licenses: the GNU GPL, the LGPL and the BSD License. They not only represent three very different styles of FOSS licensing but are also the most pervasively adopted licenses.[32] Table 1 helps us to get a quick and general overview.[33]

The FOSS licenses listed in Table 1 have the following common features:

- The source code of the original work is open.
- Making copies of the original work is allowed.
- Distribution of the original work is allowed. A copyright notice should be attached to all distributions.
- The license grant is non-exclusive, global, royalty-free, and for all purposes.
- Warranty is disclaimed.

However, these FOSS licenses differ from each other in how these rights can be exercised. For one, although authors are always required to provide access to the source code , whether redistributors are also required to provide such access varies from one license to another. For example, when redistributing a BSD-ed program, one is not required to provide the source code.

Even for licenses that require redistributors to provide the source code, there are different regulations regarding the distribution fee the redistributors can collect. The GNU GPL and the LGPL are particularly detailed about when one can collect a fee higher than the physical transmission fee. This is because the GNU GPL and the LGPL offer redistributors various ways to distribute the program, with or without the source code, while simultaneously ensuring that redistributions remain free software. An individual can sell free software for any price she wishes, since the market would help to keep the price within a reasonable range. But if a package is sold without the source code, the fee collected for the distribution of the source code itself cannot exceed the cost of physical transmission.

Clauses on derivative works vary widely. Although access to the source code of original works is a requirement, access to the source code of derivative works might not be. And even if a FOSS license requires the source code of derivative works to be open, it may not require them to be licensed under exactly the same license as the original work. For example, although a derivative work of a GPL-ed program also has to be licensed under the GNU GPL, a derivative work of a BSD-ed program does not have to be licensed under the BSD license. As a matter of fact, a derivative work of a BSD-ed program does not even have to be distributed along with source code.

FOSS licenses also differ on the possibility of allowing a FOSS program to be combined with proprietary programs. When combining different programs into a larger project, it is quite inevitable that the larger project, while embracing all or part of the source code of the programs combined, becomes the derivative work of all of the combined programs. For example, project ABC is combined with a GPL-ed program A,

[32] If we look at SourceForge.net, the largest FOSS development website, we can see that the GNU GPL, the LGPL and the BSD are the three most adopted licenses. Of the 53,026 projects that are licensed under OSI-approved licenses, 36,962 projects are licensed under the GNU GPL, 5,817 projects are under the LGPL, and 3,813 projects are licensed under the BSD. Available from http://sourceforge.net/softwaremap/trove_list.php?form_cat=14; accessed on 1 August 2004.

[33] Open Source Software Foundry is Seeking Software Freedom, A Comparison of FOSS Licenses; available from http://www.openfoundry.org/en/archives/000388.html; accessed on 2 August 2004.

Table 1

Obligations of Licensee/ License	Original Work				Derivative Work			
	As a principle, should redistributions provide source code?	When redistri-buted WITHOUT source code, can the distributor of source code alone charge a fee higher than the physical transfer cost?	When redistri-buted WITH source code, can the distributor charge a fee higher than the physical transfer cost?	Sub-licensable?	Derivative works should adopt the same license as adopted by the original work	Is source code required to be open?	Should the copyright notice of the original work be attached?	Is documen-tation required to be provided?
GNU GPL v.2	Yes	No	Yes	No	Yes, copyleft	Yes	Yes	Yes
LGPL v.2.1	Yes	No	Yes	No	Work based on the library			
					Yes, copyleft	Yes	Yes	Yes
					Executable that links a "work that uses the Library" with the library			
					No	No	Yes	Yes
BSD License	No	Yes	Yes	No	No	No	Yes	No
Artistic License	Yes	(Source code is always redistributed)	No	No	No	No	No	Yes
MIT License	No	Yes	Yes	Yes	No	No	Yes	No
Apache License v.1.1	No	Yes	Yes	No	No	No	Yes	No
Apache License v.2.0	No	Yes	Yes	No	No	No	Yes	No
Mozilla Public License v.1.1	Yes	(Source code is always redistributed)	Yes	Yes	Yes, the additional rights described in MPL may be included in an additional document	Yes	Yes	Yes
Zlib/libpng License	No	Yes	Yes	No	No	No	No	Yes
QPL	Yes	No	Yes	No	QPL requires all modifications must exist in a form separable from the original work, e.g. a Patch (does not allow peole to Qt Public modify the original work directly) and regulates the patches with clauses that are similar to the clauses other licesenses regulate the drivative works.			
					No	Yes	Yes	No
Common Public License v.1	Yes	No	Yes	Yes	No	Yes	Yes	No
Academic Free License v. 2.1	No	Yes	Yes	Yes	No	No	Yes	Yes
PHP License v. 3.0	No	Yes	Yes	No	No	No	Yes	No
Open Software License v. 2.1	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Zope Public License v. 2.0	No	Yes	Yes	No	No	No	Yes	Yes
Python Software Foundation License v. 2.1.1	No	Yes	Yes	No	No	No	Yes	Yes

a BSD program B and a proprietary program C, and has source code from all three programs. As a derivative work of program B, project ABC need not be licensed under the BSD License or even required to open its source. However, as a derivative work of program A, project ABC is required to be licensed under the GNU GPL.

Thus, the developer will have no choice but to license the whole project ABC under the GNU GPL, or find an alternative to program A, especially if she wishes to make it a proprietary software project.

This is why the GNU GPL is considered by proprietary companies to have the so-called "viral" effect, and it is regarded as unfriendly to the proprietary software development model. But the GNU GPL is designed to serve the interests of the free software community rather than the proprietary business sector. The FSF has also designed the LGPL to encourage greater use of free libraries even in proprietary software projects.

The three typical FOSS licenses – the GNU GPL, the LGPL and the BSD – are explained further below.

GNU General Public License

The GNU GPL is the classic free software license. It is also the most well-known and the most widely adopted among all FOSS licenses. The GNU GPL was developed to fulfil the freedoms defined by the Free Software Movement. It is not just a license, but is also an expression of the basic ideas behind the movement

Copyleft

- **The idea**

The way the GNU GPL guarantees freedom is also called "copyleft". While the traditional proprietary model says "copyright, all rights reserved", the GNU GPL says "copyleft, all rights reversed". Copyleft is not just about making the original work free when the copyright holder releases it, but also about keeping it free when it is being further distributed and modified. Although there is no limitation when derivative works are created only for internal use, when they are being distributed to the public, coplyleft is applied to make sure that derivative works are as free as the original work.

- **How it works**

Copyleft prevents free software from being turned into proprietary software. It uses copyright law to achieve the exact opposite of its usual purpose. Instead of being the means of privatizing software, in copyleft licenses the rights granted to authors are reversed to keep the software free.[34]

Unlike works in the public domain that everyone is free to exploit, a GPL-ed work or a copyleft-ed work is copyrighted. The author of the GPL-ed work does not give up her rights as a copyright holder, but exercises these rights in a way different from a traditional copyright holder.

If authors who want to make their software free simply disclaim their rights as copyright holders and release their work into the public domain, it will expose the work to the danger of being privatized and closed again. Instead, to keep their works and their derivates free, authors must claim their rights, and with the exclusive rights granted to them, they include the copyleft clause so as to regulate the ways other people can make use of their work. By licensing their work under the GNU GPL, authors are allowing users to have the rights stipulated by the Free Software Movement.

Also by licensing under the GNU GPL, authors require people who wish to distribute the program and developers who wish to modify the work and distribute the modified works, to take on some responsibility in keeping derivatives as free as the original work.

Major Terms and Conditions of GNU GPL

- **User's freedoms**

When a program is licensed under the GNU GPL, besides the freedom to access the source code, users are also free to:

  - Run the program (Section 0).

[34] Stallman, R., "The GNU Operating System and the Free Software Movement," p.59.

- Make copies of the program (Section 1).
- Redistribute the program, even for commercial purposes, provided an appropriate copyright notice and disclaimer of warranty are retained (Section 1). Redistribution in the object code or executable form is also possible, so long as the source code is available for all recipients (Section 3).
- Prepare and distribute derivative works of the program, provided the derivative works are also licensed to all third parties under the GNU GPL (Section 2).

- **No warranty**

Though the distribution of the work can be commercial, the work itself is licensed free of charge. Therefore, there is no warranty for GPL-ed software (Sections 11, 12). The distributor could choose to offer warranty protection in exchange for a fee (Section 1).

- **License issued directly from the author**

The work is not sub-licensable. When a program is redistributed, recipients still receive the license from the original licensor. Redistributors may not impose any further restrictions on recipients' exercise of the rights granted in the GNU GPL (Section 6).

- **Acceptance and termination**

By the act of modifying or distributing the GPL-ed program, a person indicates his acceptance of the license (Section 5). The license grant is irrevocable but when the licensee violates the license, the rights granted will be terminated automatically. However, the rights of those who received the copy of the program from her will not be affected (since they received the license from the original licensor) so long as they remain in full compliance with the license (Section 4).

- **Co-exist with other legal obligations?**

The GNU GPL does not concede to any contradictory conditions that are imposed on the recipients. If compliance with the license is not possible as a consequence of a court judgment, allegation of patent infringement or any other reason, then the recipient may not redistribute the program at all (Section 7). A GPL-ed program cannot be incorporated into a proprietary program, or linked with a proprietary library.

The full GNU GPL text can be found at http://www.fsf.org/licenses/gpl.txt

The FSF also maintains a thorough FAQ on the GNU GPL which can be accessed at www.gnu.org/licenses/gpl-faq.html

GNU Lesser General Public License

Apart from the GNU GPL, the FSF offers a special copyleft license for libraries. The GNU Lesser General Public License (LGPL) permits LGPL-ed libraries to be linked with proprietary software.

This exception is to serve different situations. It can be a strategic decision to encourage the development of proprietary applications on the GNU system.[35] For a free library whose features may be largely replaced by other proprietary libraries, releasing it under the LGPL rather than the GNU GPL can encourage its wider use,[36] and thus make possible more improvements on it. With a larger body of free software users, there would also be wider support for free software in general.[37]

However, the FSF still encourages people to use the GNU GPL for their libraries rather than the LGPL, especially for those libraries that have a significant number of unique capabilities. This is because people who are interested in utilizing such GPL-ed libraries will have to release their modules as GPL-ed software too, resulting in more useful modules and programs available in the free software environment.[38]

[35] Stallman, R., "The GNU Operating System and the Free Software Movement," p.63.

[36] Stallman, R., "Why you shouldn't use the Library GPL for Your Next Library," Feb 1999; available from http://www.fsf.org/licensing/licenses/why-not-lgpl.html; accessed on 29 May 2004.

[37] Preamble, the "GNU Lesser General Public License;" available from http://www.fsf.org/licenses/lgpl.txt;

[38] Stallman, R., "Why You Shouldn't Use the Lesser GPL for Your Next Library;" Feb 1999; available from http://www.fsf.org/licenses/why-not-lgpl.html; accessed on 29 May 2004.

Major Terms and Conditions of LGPL

The LGPL is identical to the GNU GPL in many ways, including clauses that disclaim warranty, declare that the license is issued directly from the author, and specify when the license is applied and terminated. Also, other legal obligations applied upon users are the same as those for the GNU GPL.

On users' rights, the LGPL distinguishes two different kinds of situations when one uses a library. A "work based on the Library" means either the Library itself or any derivative work under copyright law (Section 0), while a "work that uses the Library" means a program that contains no derivative of any portion of the Library but is designed to work with the Library by being compiled or linked with it (Section 5).

- **Works Based on the Library**

In the case of a "work that uses the Library", i.e., the Library itself and its derivative works, the terms are very similar to those in the GNU GPL.

- **User's Freedoms**
  - Run the program (Section 0).
  - Make copies of the program (Section 1).
  - Redistribute the program, even for commercial purposes, provided an appropriate copyright notice and disclaimer of warranty are retained (Section 1). Redistribution in the object code or executable form is also possible, so long as the source code is available for all recipients (Section 4).
  - Prepare and distribute derivative works of the program, provided the derivative works are also licensed to all third parties under the LGPL (Section 2c).

    In addition, one may opt to apply the terms of the GNU GPL instead of the LGPL to a given copy of the LGPL-ed library, especially when one is incorporating part of the code into a program that is not a library (Section 3).

- **Works that use the Library**

In the case of a "work that uses the Library," the work itself is NOT subject to the LGPL. But when linking the "work that uses the Library" with the Library, an executable version that is a derivative work of the Library would be created, and such a version is covered by the LGPL (Section 5).

Although the LGPL allows authors to distribute the object code of executables (Section 5) and license these under terms of their choice, it is also required that those terms permit modification of the work for the customer's own use and reverse engineering. When distributing executables, the author has a choice either to distribute the Library together, provided the source code of the Library is made available in those ways similar to the distribution of GPL-ed programs, or not to distribute the Library together but only use a suitable shared library mechanism to link with the Library (Section 6).

By creating this category, the LGPL provides a way for LGPL-ed libraries to be used in proprietary programs. The full LGPL text can be found at www.fsf.org/licenses/lgpl.txt.

BSD Style Licenses

The Berkeley Software Distribution (BSD) License was first used for the Berkeley Software Distribution, a version of Unix developed in the University of California at Berkeley.[39] It is easy to follow the BSD License template to create one's own license by changing the values of owner, organization and year, which appear in the copyright notice and the license. Unlike the GNU GPL and the LGPL, BSD style licenses are not copyleft licenses. A BSD License allows people to freely distribute and modify the original work, but it does not require that the modified works be as free as the original work. BSD style licenses are relatively simple and have only limited restrictions on the use of the software.

[39] "MIT License Definition", June 2004; available from http://www.bellevuelinux.org/mitlicense.html; accessed on 1 July 2004.

- **User's Freedoms**
  - Make copies and redistribute the program, with either its source code or its binary code. The distributor is not obliged to provide the source code.
  - Prepare derivative works and distribute them, either with their source code or binary code. The author is free to choose either FOSS or proprietary licenses for derivative works.
  - Incorporate the program into proprietary software.

The original BSD License (four-clause BSD) has an advertising clause. The revised BSD License (three-clause BSD) is very similar to the MIT License, but the latter does not have the "no endorsement for derivative works" clause. There is also the two-clause BSD, which has taken away the endorsement clause and is most similar to the MIT License.[40]

Multiple Licensing

It is important to note that a work can be licensed under more than one license. The choice of license reflects the type of relationship that the author wishes to have with the user of the copyrighted work. Since there can be more than one kind of user, and more than one possible relationship, the copyright holder is entitled to choose different kinds of licenses for different situations.

Take OpenOffice as an example. OpenOffice is dual-licensed under the GNU GPL and the Sun Industrial Standards Source License (SISSL). Although OpenOffice states clearly that users are encouraged to use the GNU GPL to participate fully in the OpenOffice community, SISSL is provided as an alternative for developers and companies who are not able to use the GNU GPL.[41 ,42] MySQL is another example. MySQL offers both the GNU GPL and a commercial license. Organizations that do not want to release their applications under the GNU GPL can choose to use MySQL under the MySQL Commercial License.[43]

What about Documentation?

GNU Free Documentation License (GNU FDL)

Good documentation and manuals are extremely important for FOSS programs. When they are not licensed as free/open, it is difficult for people to make complete use of relevant FOSS programs.

Although the GNU GPL is a license designed mainly for software, it can also be used for works that are not software, so long as it is defined clearly what the "source code" is when adopting the license.[44] The FSF also provides a license that is specially designed for documentation. The GNU Free Documentation License (GNU FDL or FDL) is a form of copyleft license for manuals, textbooks or other documents that grants everyone the freedom to copy and redistribute the documents, with or without modifications, either commercially or non-commercially.[45]

By applying GNU FDL to a document, the author grants users the right to make verbatim copies of the work, to modify the work and to distribute modified works. Since it is a copyleft license, it requires the copy and the distribution of modification of the FDL-ed work is also licensed under the FDL.

[40] "WikiReader, Free Software and Free Content," June 2005; available from http://upload.wikimedia.org/wikipedia/en/a/a9/WikiReader_Free_Software_and_Free_Contents.pdf; accessed on 8 July 2004.

[41] "Licenses," August 2002; available from http://www.openoffice.org; accessed on 28 June 2004.

[42] One is entitled to choose the license only for the code that belongs to him. In cases involving collaboration among different individuals or entities, all of the co-contributors have to agree on which licenses to choose for the work as a whole. This can be done either by an agreement among all co-contributors or, as in the OpenOffice case, participants in the project are required to sign a Joint Copyright Assignment with Sun Microsystem.

[43] "MySQL Licensing Policy;" available from http://www.mysql.com/company/legal/licensing/; accessed on 10 November 2004.

[44] Available from http://www.gnu.org/licenses/fdl.html; accessed on 4 August 2004.

[45] Available from http://www.gnu.org/licenses/licenses.html#TOCFDL; accessed on 4 August 2004.

Table 2

	Attribution required	Allow commercial uses	Allow derivative works	Derivative works should be licensed under the same license as the original work is
CC BY	Yes	Yes	Yes	No
CC BY-NC	Yes	No	Yes	No
CC BY-NC-ND	Yes	No	No	
CC BY-NC-SA	Yes	No	Yes	Yes
CC BY-ND	Yes	Yes	No	
CC By-SA	Yes	Yes	Yes	Yes
CC NC*	No	No	Yes	No
CC NC-ND*	No	No	No	
CC NC-SA*	No	No	Yes	Yes
CC ND*	No	Yes	No	No
CC SA*	No	Yes	Yes	Yes
GNU FDL	Yes	Yes	Yes	Yes

BY: Attribution. For any reuse and distribution, it is required that credit is given to the original author.
NC: Non Commercial. The work cannot be used for commercial purposes.
ND: No Derivative Works. The work cannot be altered or transformed; derivative works built upon the work are not permitted.
SA: Share Alike. It is allowed to alter and transform the work, and to prepare derivative works upon the work, so long as the resulting work is licensed under a license identical to this.
** Starting in 2004, Creative Commons made the "attribution" requirement the default in the second version. Thus only the first six CC licenses above remain in the second version.*

Creative Commons Licenses

Inspired by the FOSS development, the Creative Commons advocates for openness of digital content and is urging for a more reasonable and flexible layer of copyright in the face of increasingly restrictive default rules.[46]

In 2002, the first versions of Creative Commons Public Licenses (CC licenses) were released. By identifying major concerns of authors – i.e., whether attribution is required (attribution, BY), whether users are allowed to make commercial uses of the work (non-commercial, NC), whether users are allowed to make derivative works (no derivative works, ND), and when derivative works are allowed, whether they are required to be licensed under exactly the same license as the original work (share alike, SA) – Creative Commons developed a set of 11 different CC licenses. Each represents a unique combination of the above four conditions. Authors are free to choose among the 11 licenses and decide which best suits their needs and works.

In 2004 Creative Commons released the second version of CC licenses. Since the requirement of attribution has been widely adopted by users of CC licenses, the attribution requirement has become default, and thus there are only six CC licenses in the second version. However, the 11 licenses in the first version are not superceded and are still available (Table 2).[47]

CC licenses are designed for all kinds of digital content except for software, including art works, photographs, music and literary texts. CC licenses do not deal with the FOSS issue, since ideas in the works referred to are transparent and are not compiled into forms that cannot be perceived. Some CC licenses do not allow modification and might not be regarded as "free". However, CC licenses are successful in spreading the idea of freedom and openness to the greater public, which might not be familiar with recent software developments fostered by FOSS movements.

[46] Creative Commons, "Some Rights Reserved, Building a Layer of Reasonable Copyright;" available from http://creativecommons.org/learn/aboutus/; accessed on 4 August 2004.
[47] Creative Commons Public Licenses are available at http://creativecommons.org/licenses/.

SCENARIOS

Different stakeholders will have different uses for FOSS. A developer might use a program more intensively than an end-user, which means that the developer's activities might be subject to more restrictions than an end-user. The following section tries to provide some scenarios as examples to explain the different legal issues that may arise for different stakeholders.

End-user (Individual/Business/Government)

> *Abul is a public high school teacher. His school cannot afford expensive license fees for proprietary office applications. Although proprietary software companies offer special rates for schools, Abul wanted to find an alternative solution to reduce students' dependence on proprietary software. His friend, Nazlee, a programmer who has participated in FOSS projects, introduced him to a FOSS office application. Abul and his colleagues then downloaded FOSS office solutions and taught students both proprietary and FOSS applications. He is satisfied with the performance and introduced Nazlee's program to his colleagues. Gradually, the school administrative body began to use the FOSS solution for administrative work.*

In this case, neither Abul (an individual) nor his school (a public government body) made any modification to the software that they downloaded from the Web. They were simply end-users.

The situation for end-users is relatively simple. The end-user of a software program may be an individual, a government body, or a business entity. These individual persons or legal entities may have different reasons to use FOSS. Some may be trying to find a cheaper solution or a solution that suits their needs better; others may wish to use FOSS for better customization; still others may wish to reduce their dependence on proprietary companies.

- **Legal issues involved**

  The way end-users use FOSS solutions might not be very different from the way they use proprietary solutions. They download a copy of a FOSS solution or purchase a copy (usually in exchange for some support and services), install it in the computer (thus making a copy on the hard disk), run the program, and have its functions serve their needs. The rights that are of concern here are the right to make copies of the program and to run it. (The act of running a program may also count as an act of making copy, but it is stated differently in some FOSS licenses. For example, the GNU GPL has no restrictions on the running of the GPL-ed program but does regulate the act of making a copy.) These rights are granted by all FOSS licenses. Thus there are fewer legal disputes involving end-users. However, end-users do need to consider some issues.

  - Issue 1: Technical Support

    Since an end-user might not be a computer whiz, she might have concerns regarding technical support when choosing a FOSS solution. Thus, instead of simply downloading a copy of the FOSS solution for free, she might choose to purchase a box of FOSS in a shop where a proprietary solution is also available, sometimes at approximately the same price. The difference is that when purchasing, let's say, a commercial Linux distribution in the store, the end-user is not paying for the license fee, but for the service and support. When the term of service expires, the end-user can choose to pay for another term of service, or ask other available providers for similar services.

  - Issue 2: Customization

    When existing FOSS solutions do not fit their needs, end-users might need to ask individual developers or vendors to customize the solutions. In such cases, since end-users may not be technically savvy enough to detect possible infringements, they may wish to have a written clause in the contract ensuring that the vendor or developer will take on the entire responsibility for any possible copyright infringement, and will compensate for any possible losses that may be caused by allegations of infringement. The buyer is free to add these clauses to the contract when negotiating with the vendor or developer.

- Issue 3: Government Procurement
  Since FOSS licenses are different from traditional software licenses, governments should be particularly aware of the differences when they open a bid for software solutions or sign a contract with vendors. Existing government bid and contract templates may have been drafted under the traditional proprietory model of traditional copyright law, and have to be examined, or revised, if they fail to treat FOSS and proprietary software equally.

Developer (Individual, Business)

Developers (individuals and business entities) need to be more careful with the terms and conditions of different licenses while using FOSS. Developers usually not only run and copy the software, but also create derivative works from the software, and distribute these derivative works together with the original program. Therefore, for developers to contribute to the development of a certain FOSS program, it is essential to have the rights to run the program, to make copies, to distribute the program and to prepare derivative works.

These rights are granted by all FOSS licenses, for these essential rights are considered important both in the Free Software Definition and the Open Source Definition. Nevertheless, different FOSS licenses may have different restrictions on exercising these rights, especially on creating and distributing derivative works. Developers should pay particular attention to this, and consult their lawyers on their specific situation when needed.

There are different considerations when a developer participates in different stages of software development.

When Starting a New Project

Abul's colleague Jolly is the school librarian. The school library is not that large, but it is open to villagers. Jolly sought her friend's help to write a program that would enable her to keep an accurate record of the books in the library.

- **Legal issues involved — choosing a license of one's own**

Developers: What does this project mean to me and to others? How do I want others to be involved? What do FOSS licenses say? What are the differences between FOSS licenses?

The situation is relatively simple if the developer is starting a new project without using any existing modules, since she will not have to look through the licenses of existing modules that she might have used.

However, starting a new project is not an easy task either. The different characteristics of the FOSS license she chooses will have a significant influence on the possible development path of the project. The developer should define her main concerns before choosing a license.

For example, if the developer is a supporter of copyleft, she may stick with the GNU GPL or the LGPL. If the developer thinks she doesn't need to require people to license their modified works under FOSS licenses, a BSD-style license would be appropriate. Or when the developer thinks it is better to control the development in a firm and central line, she might not be interested in BSD style licenses. But if forking is preferred in the future development, BSD style licenses may be a better choice.

Developer: Can I change my mind after licensing my project?

The copyright holder of a project can always decide to choose another license for the program, even when the previous versions have already been licensed under certain FOSS licenses. This will not affect the rights of the recipients of the previous versions since license grants are irrevocable. The situation will be more complicated if contributions from the community have been incorporated into the newer version, which means that these other contributors may claim copyright to certain pieces of the code in the newer version. In this case, unless there is prior agreement, the license must be chosen by all contributors.

Developer: I don't like any of the existing FOSS licenses. Can I start a new one?

Though there are already many FOSS licenses, it is possible that a developer will find that she does not like any of the available licenses. As long as the developer owns the code, she is entitled to choose any license for the project, including a new one that she drafts by herself. However, creating a new FOSS license requires legal knowledge and skill to avoid vagueness and loopholes. Also, there are already many FOSS licenses and the transaction cost for understanding these licenses is high. Creating a new license is not recommended unless a developer has strong reasons to do so.

When Modifying an Existing Module

The office application Abul and his school are using has an English interface. It does not support the local language. Using an English interface might not pose a problem for high school students. However, it is difficult when Abul tried to teach villagers. He consulted Nazlee about the problem. Nazlee has constantly contributed to FOSS programs and is also quite familiar with the source code of the office application. She discussed this with a few friends and, as a team, they began to localize the application.

- **Legal issues involved: Ascertain the license of the program to be modified**

When a developer tries to modify an existing module, and when the modification is not solely for her own use but for further distribution (e.g., localizing a project), she needs to first identify the license of the module.

Developers: Under the license, what are the rights I am granted and what are the restrictions in exercising those rights?

For example, on the distribution of a FOSS work, some FOSS licenses (e.g., the GNU GPL, the LGPL) may require distributors to provide both object code and source code, or at least provide the information on how to access the source code. On modifying a FOSS work, some FOSS licenses (e.g., GNU GPL, LGPL, BSD) may require the modifier to provide documentation of the changes being made. On distributing the derivative work, copyleft licenses require derivative works to be licensed under the same license as the original work, while other FOSS licenses allow the modifier to choose a different license (BSD, MIT).

If Nazlee and her friends are trying to localize the dual-licensed OpenOffice, and they decide to use the one under the GNU GPL, then the localized OpenOffice would also be GPL-ed.

Some FOSS licenses (e.g., the MIT License) may allow users to sublicense the original work. This means that when distributing the verbatim copy of the original work, within the scope granted by the original copyright holder, the distributor may choose a different license and become a licensor him/herself. In such cases, when a developer creates a derivative work and distributes it together with the original work, he/she can choose to become a licensor of both the original work and the derivative work, which simplifies the legal relations to one that exists only between the two parties. If a sublicense is not allowed, people who receive the modified work would have two licensors for this piece of work. The licensor of the original work will be the author of the original work, while the licensor of the derivative work will be the developer who prepared the derivative work.

When Integrating Different FOSS Modules into One Service

Nazlee works in AA Software Inc. To better oversee the many different projects that the company is developing, the team built a project management system by integrating different FOSS modules. The management system is only for internal use now, but since it is pretty handy, the company also plans to distribute it commercially in the future.

FOSS licenses do not impose restrictions on modifications that are made for internal use. But when a public distribution is created based on these modifications, the developer must consider all the licenses of the modules being used.

- **Legal issues involved: Identify the licenses of the programs being integrated, and see if these licenses are compatible.**

When integrating several different modules, the developer may end up modifying these modules. Therefore, it is essential to ascertain the licenses of each module. If they happen to be licensed under the same license, such as the GNU GPL, then the integrated system would be licensed under the GNU GPL. The situation is similar when all modules are licensed under the BSD License. But in this case, AA would be able to choose another FOSS license or a proprietary license for the modified modules and the integrated system.

However, if some of the modules have different licenses, then AA will have to look at the compatibility of these licenses. When two licenses are compatible, the two modules licensed under the two licenses can be combined into a larger work while complying with both licenses.[48]

When combining a GPL-ed program and BSD-ed (GPL-compatible) into a larger program, the larger program will have to be GPL-ed to meet both the requirement of the GPL-ed program and BSD-ed program. When some of the modules are GPL-ed but other modules are GPL-incompatible. In this case, AA must decide which module is more important for them and replace the other one with a module with a compatible license.[49]

The licenses used in different modules and the way they are combined together would determine how the integrated system can be licensed and distributed.

- **Other considerations – choice of law and choice of venue clauses**

Finally, for those who are able to choose licenses for their programs, either because they started their own programs or they are allowed to choose licenses for the derivative works they prepared, they should be aware that many OSI-approved licenses are developed by proprietary software companies. Some are designed to meet their company policy and strategy, and thus might not be a good choice for developers in general. Some technical issues, such as clauses on choice of law and of venue (which could be found in the Qt Public License, the Mozilla Public License, the Common Public License, etc.), may become significant when a lawsuit is brought up.

Vendor/Producer (Business)

Nazlee and her friends have made the localized version of the FOSS office version available. AA Software Inc. is interested in this application. They have also developed some other small but useful programs for administrative work. They package the localized office together with their own programs (licensed under their proprietary license). The package is a big hit. A few months later, AA also decides to commercially distribute the project management system that they had integrated from different FOSS modules.

Mere distribution

In this situation, both FOSS and proprietary programs are distributed in one package. For the FOSS application, AA is merely a distributor, and must distribute it as its FOSS license requires. For proprietary programs, AA holds the copyright and is able to choose the license and ways of distribution. It is all right to put FOSS and proprietary applications into one distribution package, such as one CD-ROM, if the applications function separately and do not link together to create any derivative work.

Distribution of integrated systems

In the case of an integrated system distribution, what is key are the licenses of the different integrated modules and the ways in which the modules are combined. As explained earlier, AA needs to first make sure that the licenses of the different modules allow AA to combine them. These licenses will also determine the ways by which AA can distribute the integrated system.

Other considerations – drivers and certifications

[48] "What does it mean to say that two licenses are compatible?;" available from http://www.gnu.org/licenses/gpl-faq.html#WhatIsCompatible; accessed on 7 July 2004; "FAQ on Open Source Licenses;" available from http://www.openfoundry.org/en/archives/FAQonOSL.pdf; accessed on 7 July 2004

[49] "Various License and Comments About Them;" available from http://www.fsf.org/licensing/licenses/index_html#GPLCompatibleLicenses; accessed on 7 July 2004. The FSF provides a list of GPL-compatible and GPL-compatible FOSS licenses.

One difficulty that FOSS vendors might encounter is that hardware vendors may not be aware of FOSS software and thus fail to provide drivers that will enable FOSS applications to work on the hardware. It is important to promote the idea of FOSS among hardware vendors. This will be easier when there is a larger group of FOSS users.

Likewise, certification is sometimes needed for FOSS to work properly with specific proprietary software. A larger FOSS user group will encourage proprietary software companies to certify FOSS applications that might be used together with their programs.

Other considerations – FOSS used in embedded systems or devices
FOSS is also used in embedded systems in electronic devices, such as cell phones, hand-held devices, digital cameras and DVD players. The use of FOSS may help device manufactures to lower their cost when developing new products. The distribution of the device is different from the distribution of the FOSS itself. With regard to the latter, the rules of specific licenses still apply.

Government-sponsored Projects

FOSS movements and rapid FOSS developments have received attention not just from the FOSS community, but also from academics and policy-makers. In some Asian countries, governments work with PC manufactures/vendors to provide affordable PCs bundled with FOSS operating systems and office applications.[50] Governments also support FOSS development, generate FOSS-related projects and promote FOSS as a national technology and industrial policy.[51] But long before governments began to notice the potential of FOSS and developed a clear position on it, some government-affiliated academic institutes have already been working on FOSS-related projects.

The FSF-maintained FAQs about the GNU GPL also list questions about whether the United States Government could release a program under the GNU GPL or release improvements to a GPL-ed program.[52] Situations may differ from country to country and from case to case under different government regulations in different countries. Most government regulations on government-sponsored projects are usually drafted under their domestic copyright and patent law and might be informed by a more protectionist mentality and thus be unfamiliar, or even unfriendly, to FOSS licensing and development models.

Below are two cases of government-funded FOSS studies. The first one is about FOSS-related studies made in a government research institute without related government policy, while the second one is about a national FOSS project.

Government-funded FOSS Projects: Cases from the Asia-Pacific

A FOSS Project under a Government-affiliated Research Institute: Multi-Lingual Editor, Japan

Emacs is a multilingual text editor first developed by Richard Stallman at MIT. After the GNU project started in 1984, the development of GNU Emacs was started and it was first released in 1985, under the GNU GPL.

The Japanese governmental research institute, Eletrotechnical Laboratory (ETL),[53] began to work on the multilingual information processing and integration of GNU Emacs and Mule (multilingual text editor based on Emacs and later merged into GNU Emacs as MULE) in the mid-1990s, but there were various copyright issues.

ETL was a government research institute, and the licensing model in the GNU GPL is very different from Japanese copyright law, so no one was able to decide whether ETL could assign the code to the FSF and

[50] "Malaysian 'People's PC'- Microsoft experience "Thailand Linux" pain all over again," Mar 2004; available from http://www.asiaosc.org/article_191.html; accessed on 7 July 2004. See also Koanantakool, T., "A Case for Nation-wide PC Distribution," November 2003; available from http://www.asia-oss.org/; accessed on 7 July 2004.

[51] Related links available at uwstudent.org/wiki/OpenSourceInGovernment; accessed on 8 July 2004. Also available from http://www.asiaosc.org/enwiki/page/Ideas_for_OSS_policy.html; accessed on 8 July 2004.

[52] Available from http://www.fsf.org/licensing/licenses/gpl-faq.html#GPLUSGov and http://www.fsf.org/licensing/licenses/gpl-faq.html#GPLUSGovAdd; accessed on 10 July 2004.

[53] Handa, K., "Development of Multi-Lingual Editor," 2003; available from http://www.asia-oss.org/nov2003/present/handa/handa.html; accessed on 10 July 2004.

release the code under the GNU GPL. As a result, ETL never officially released the code but released the trial versions instead. More negotiations between ETL and FSF took place later, and resulted in a special agreement. The FSF agreed not to require ETL to assign the copyright of the modified code to the FSF, and ETL agreed to grant FSF the right to use the code. This was the first time that part of the code in Emacs did not belong to the FSF.

In 2001, ETL was reorganized into the National Institute of Advanced Industrial Science and Technology (AIST). Although AIST is still a government-funded institute, it is an independent organization and its assets are not national property. It seemed that AIST would be able to release the code under the GNU GPL officially. But, initially, it was still very difficult for the higher levels of AIST to make a final decision. It took them another year of internal negotiation to decide that AIST was entitled to release their works and choose the licenses of their works. It was also not easy to convince people about the advantage of adopting the GNU GPL. According to Dr Kenichi Handa, a senior researcher in AIST, it was never clear what convinced the AIST management to make the final decision.

This happened before the Japanese Government had formed a clear position on FOSS development. During an open source conference among Asian countries in 2003 where Dr Handa was invited to give a talk on the development of Emacs, Shuichi Tashiro, the leader of the Japanese FOSS project under the Ministry of Economic, Trade and Industry, said that the Japanese Government has made necessary regulatory revisions to give developers of government-funded projects the copyright (and thus the right to choose the license) so long as the law was applicable from the beginning of the project.

In this case, we can see that when the government is not familiar with the FOSS licensing and developing model, related government regulations may create unnecessary difficulties for government-affiliated research institutes seeking to participate fully in FOSS development. It took AIST, formerly ETL, years to finally be able to officially release the code under the GNU GPL. Even though now there is a special regulation to facilitate the use of FOSS licenses for government-funded open source projects, as they are still considered exceptions.

A National FOSS Project: Free Software Industrial Development Project, Taiwan

Under pressure from Congress, the Taiwanese Government began the planning of a national FOSS project in 2002, and in 2003 a significant budget had been allocated to a five-year FOSS project. The Ministry of Economics Affairs (MoEA) was assigned to structure, sponsor and oversee all of the sub-projects.

Under general government regulations administered by the National Science Council (NSC), although the results can be copyrighted by the entity which carries out government-funded projects, applications of such results are still subject to certain regulatory principles. Unless it would be more beneficial for the national development of science and technology, the results have to be:

- Licensed for a fee.
- Licensed to Taiwanese institutes or firms.
- Used or manufactured within Taiwanese jurisdiction.

Though exceptions might be made for FOSS projects, the law had not been officially interpreted in this way, and no one wanted to risk violating the regulation.

In addition, the national FOSS project was assigned to the MoEA, which has the more important task of protecting national interest and economic competitiveness. Thus their regulations are more protectionist/restrictive than the general rule. These restrictive regulations were applied to the national FOSS project. Under MoEA regulations, only the third principle (used and manufactured within Taiwanese jurisdiction) can be exempted. This meant that the outputs of the national FOSS project had to be licensed for a fee, and it can be licensed only to Taiwanese institutes or firms. Such principles are inconsistent with the FOSS licensing model, making it difficult for all sub-projects under the national FOSS project to release their code.

This issue was raised as soon as the five-year FOSS project started. Different government bodies met several times to find a solution. Because the FOSS licensing model was so alien to the models they were used to, the problem was not solved until mid-way into the second year (2004) and the code developed in the first year was not officially released in time.

This was particularly problematic since one of the sub-projects under the national FOSS project was integrating an existing FOSS program. At the same time, the sub-project intended to participate in and contribute to this particular FOSS project. When the community was about to incorporate all of the recent developments and release its newer version under the GNU GPL in March 2004, they found it difficult to incorporate the code developed under the government-funded sub-project in Taiwan.

It was not until after a negotiation held in May 2004 that different government bodies finally came out with a solution. The MoEA submitted the case to the Administrative Yuan (highest administration body) to obtain an official interpretation from the Government regarding whether FOSS projects meet the exception clause and are thus exempt from the principles. Meanwhile, the MoEA began to look into the possibility of revising its restrictive regulations.

The official interpretation was finally made by the Administrative Yuan in July 2004, 18 months after the official launch of the national FOSS project. Under the new interpretation, government-funded FOSS projects met the exception clause of the general NSC rule and can be exempted from the principles that conflict with FOSS licenses. Although the MoEA regulation has not yet been modified, some code-generating FOSS projects are assigned to NSC and the general NSC rule, rather than the more restrictive MoEA rules . It is hoped that FOSS projects will be able to release the code under FOSS licenses thereafter.

This case shows that while the government has started to recognize the importance of FOSS development, its regulatory and administrative structure might not be ready to accommodate FOSS. In the case of the Taiwan National FOSS Project, with the combined efforts of related government and project personnel, the problem was finally solved to a certain extent. But it had already caused some serious problems, especially in collaborating with international and local FOSS communities. As many countries now also recognize the importance of FOSS development, and are starting or planning to start their governmental FOSS projects, it is critical that the related legal structures are examined and updated to facilitate FOSS development.

ONLINE LEGAL RESOURCES AND MATERIALS

- Free Software Foundation, http://fsf.org
- Open Source Initiative, http://opensource.org
- Open Source License Law Resource Center, http://www.denniskennedy.com/resources/technology-law-central/opensourcelaw.aspx
- Open Source Licensing, http://www.anu.edu.au/people/Roger.Clarke/EC/OSLic.html
- WikiReader, Free Software and Free Content, http://en.wikipedia.org/upload/a/a9/WikiReader_Free_Software_and_Free_Contents.pdf
- Groklaw, http://www.groklaw.net/index.php
- FLOSS Concept Booklet, http://www.sarai.net/floss_book.pdf
- Frequently Asked Questions about GNU GPL, http://www.gnu.org/licenses/gpl-faq.html
- Quiz to Test Your Knowledge of the GPL and LGPL, http://www.gnu.org/cgi-bin/license-quiz.cgi
- Apache License and Distribution FAQ, http://www.apache.org/foundation/licence-FAQ.html
- Mozilla Relicensing FAQ, http://www.mozilla.org/MPL/relicensing-faq.html
- Netscape Public License FAQ, http://www.mozilla.org/MPL/FAQ.html
- Feature, Open Source Families and Facts, http://www.unixreview.com/documents/s=8925/ur0312b/
- A Comparison of Open Source Licenses, http://swan.iis.sinica.edu.tw/LicenseWizard/OSI_License_compare_v3.0.4EN.pdf
- Electric Frontier Foundation, http://www.eff.org
- Foundation for a Free Information Infrastructure, http://www.ffii.org
- IP Justice, Campaign for an Open Digital Environment, http://ipjustice.org/CODE/
- League for Programming Freedom, http://lpf.ai.mit.edu
- Infochange - Intellectual Property Rights, http://www.infochangeindia.org/Intellectual_Property_Rights.jsp
- Copyrights and Software Protections by Patents and Copyrights, http://www.ladas.com/Patents/SoftwareProtectionIndex.html
- Journal of Information, Law and Technology, http://elj.warwick.ac.uk/Jilt/
- Creative Commons, http://creativecommons.org
- Open Source Legal Toolkit, Massachusetts Government Information Technology Division, http://www.mass.gov/portal/site/massgovportal menuitem.769ad13bebd831c14db4a11030468a0c?pageID=itdsubtopic&L=5&L0=Home&L1=Policies%2c+Standards+%26+Legal&L2=Legal+Guidance+%26+Documents&L3=Software+Licensing+%26+Development&L4=Open+Source+Legal+Toolkit&sid=Aitd
- Software Freedom Law Center, http://www.softwarefreedom.org/index.html

GLOSSARY

This section is a derivative work of its Mandarin version co-authored by Rong-chi Chang and Chingyuan Huang, former colleagues of the OSSF, Institution of Information Science, Academia Sinica.

- **Copyleft**

  Proposed by free software advocates, copyleft is an alternative framework conceived within copyright law which usually confers exclusive rights to copyright holders and thus limits access to the work by all others. Authors may want to "copyleft" their works to grant certain rights to people who are interested in distributing or modifying their works, provided these people will also "copyleft" all the derivative works. Although copyright and copyleft might represent very different ideas regarding the relationship between authors and their works, copyleft is not against copyright law. On the contrary, without the rights granted by the copyright law, authors will not have the power to copyleft their works. Please also refer also to the definition provided by the Free Software Foundation at http://www.gnu.org/copyleft/copyleft.html.

- **Copyright**

  A bundle of rights regarding the use of a creative expression (including literary works, music compositions, movies, paintings, software, and the like) that the law grants exclusively to the author. Copyright is applied to a work upon its creation. Except for the limitation set by copyright law, any use of a work without the copyright holder's consent is regarded as an infringement. Note that copyright law protects only the expression of ideas but not the ideas themselves.

- **Copyright Holder**

  The individual or legal entity who is entitled to exclusive rights under copyright law. It is usually said that copyright law aims to protect authors of creative works. But since most of the rights protected are treated as property rights and may be transferred, many copyright holders are not the authors of the works themselves but their employers or those who have commissioned these works.

- **Derivative Work**

  Copyright law is applied to every work once it is created. With the consent of the copyright holder, one can use this (original) work to create derivative works. For example, a newer version of a program might contain all or part of the code of the earlier version. Thus the newer version is a derivative work of the earlier version. Translation of a document is also regarded as a type of derivative work.

- **Distribution/ Redistribution**

  Distribution of the copies of a work is also an exclusive right granted to the copyright holder. In FOSS licenses, all receivers of copies of a program are allowed to make further distributions. The term redistribution may be used when emphasizing that the distributor has received the program from somewhere and is distributing further.

- **Fair Use**

  Copyright law seeks to maintain a balance between private and public interests. "Fair use" is developed to limit excessive copyright protection and to allow the general public greater access to copyrighted works. When a work is used without the consent of the copyright holder for purposes of criticism, comments, news reporting, teaching, scholarship or research, such use might not be considered an infringement. Though copyright may differ in different jurisdictions, usually the following factors are considered by the court in deciding whether a case falls under fair use or is an infringement:

  - The purpose and character of the use, including whether such use is of a commercial nature or is for non-profit educational purposes.
  - The nature of the copyrighted work.
  - The amount and substantiality of the portion used in relation to the copyrighted work as a whole.
  - The effect of the use upon the potential market for or value of the copyrighted work.

- **First-sale Doctrine**

  The first-sale doctrine is an exception of copyright law that is codified in Section 109 of the US Copyright Act. Similar doctrines may be also be adopted by other countries. The doctrine allows the person who purchased a legally acquired copy of a copyrighted work to further distribute (including sell, rent or give away) the copy without permission from the copyright holder. But the first-sale doctrine does not apply to phono-records and computer software.

- **License**

  A license is a legal document that copyright holders may adopt to regulate how people can use their works. Users are often required to accept the terms and conditions of a license as a prerequisite to their use of the copyrighted works.

- **Multiple Licensing**

  The copyright holder of a work can have various ways of making use of his/her work available to others. The terms and conditions she would want users to accept may differ from case to case. For example, the copyright holder of an editor software may be willing to issue an academic license that is cheaper and more affordable for students, while commercial licenses are adopted when the program is sold to commercial entities. A copyright holder can also decide to license a work under both FOSS licenses and proprietary licenses to achieve different purposes.

- **Public Domain**

  The term public domain is used to describe all creative works that are not protected by copyright law and can therefore be used freely. Works that are in the public domain might be cultural heritage that came to existence before copyright law, or works that were once protected but whose copyright has expired, or works for which their copyright holder decides not to claim copyright. In the latter case, the disclaimer must be made explicitly. In some countries, a signed written document deposited with a national registrar may even be required. Works that are licensed under FOSS licenses are still copyrighted and do not fall into this category.

- **Source Code**

  Source code is written in special kinds of languages designed for programming. A program in its source code form might not be easy for lay people to understand, but it is comprehensible to trained programmers. When the source code is converted to machine readable form, even programmers will have difficulty understanding and modifying the program. Therefore, access to the source code is a prerequisite for the development of FOSS and a principle embraced in all FOSS licenses. A more detailed explanation of "source code" can be found in the Glossary of the introductory primer, *Free/Open Source Software, A General Introduction*, which is available online at www.iosn.net/downloads/foss_primer_current.pdf.

- **Sub-license**

  When a copyright holder licenses her work to someone else, she can also choose to allow the licensee to sublicense the work. That is, when the licensee distributes the work, within the scope of rights granted by the licensor, the licensee is not only a (re)distributor but also a licensor of a sub-license between her and the other party (licensee of a sub-license). However, most FOSS licenses do not grant people the right to sub-license. For example, A is the copyright holder of X program. B receives a copy of X and distributes more copies. C receives the copy from B. If A does not grant B the right to sub-license, both B and C receive the license directly from A. If A grants the right B to sub-license program X, within the scope of the rights granted by A, B may start a new license and him/herself become a (sub)licensor of program X.

- **Warranty Disclaimer**

Warranty is a guarantee made by the vendor against potential liabilities arising from the use of a product. All FOSS licenses come with a warranty disclaimer. Such clauses are designed to protect the author of FOSS programs, for these programs are licensed without royalty and changes might be added in its development. However, although FOSS programs themselves are royalty-free and disclaim warranty, vendors of FOSS programs can always provide their customers with a warranty and various kinds of supports for a fee.

Another reason why licenses of community distributions may not include warranty clauses may be because developers are expected to understand the code and fix the bugs, and are invited and expected to take on some responsibility as a member of the community. However, in commercial distribution, it is unreasonable to expect customers to be capable to read and change the code. In some countries, failure to provide minimum guarantee will lead to consumer protection issues

ABOUT THE AUTHOR

Shun-ling Chen did her master degrees in law both at National Taiwan University and Harvard Law School. Her research interests and political commitments have been mainly about how self-organized social agencies are able to instigate driving forces for structural transitions. She has been working with various NGOs and sees FOSS as one of such community building processes. Until the summer of 2004, she was the project co-lead of Creative Commons Taiwan and served in the OSSF[54] project as the project manager of its Law and Policy team.

Built with FOSS modules, OSSF is a public platform that offers free (as in free beer!) tools and spaces for FOSS community to develop their projects. The OSSF is carried out by the Institute of Information Science, Academia Sinica, Taiwan.

[54] OSSF is a recursive acronym for "OSSF Supports Software Freedom", http://www.openfoundary.org

ACKNOWLEDGEMENT

The author would like to acknowledge her debt of gratitude to the following:

- The reviewers of this primer, Richard M. Stallman, Eric S. Raymond, Dr. Nah Soo Hoe, Aniruddha Shankar, Mahesh T. Pai, and Kenneth Wong for their valuable comments and suggestions.

- Her colleagues at the OSSF project and the FOSS community members who have developed their projects on the Openfoundry, for providing their valuable inputs towards understanding FOSS licensing issues and developing the ideas contained in this primer.

- The current and former UNDP-APDIP colleagues Sunil Abraham, Khairil Yusof, Phet Sayo, Shahid Akhtar and Kenneth Wong for their efforts and assistance that made this primer possible.

- And finally, those who have contributed their time, efforts, and resources to the FOSS movement with enthusiasm which has shown us what voluntary collaboration among human beings can achieve.

APDIP

The Asia-Pacific Development Information Programme (APDIP) is an initiative of the United Nations Development Programme (UNDP) that aims to promote the development and application of information and communication technologies for sustainable human development in the Asia-Pacific region. APDIP aims to meet its goals by focusing on three inter-related core areas: (i) policy development and dialogue; (ii) access; and (iii) content development and knowledge management.

APDIP collaborates with national governments, regional, international and multi-lateral development organizations, UN agencies, educational and research organizations, civil society groups, and the private sector in integrating ICTs in the development process. It does so by employing a dynamic mix of strategies – awareness raising, capacity building, technical assistance and advice, research and development, knowledge sharing and partnership building.

www.apdip.net

IOSN

The International Open Source Network (IOSN) is an initiative of APDIP and supported by the International Development Research Centre of Canada. IOSN is a Centre of Excellence for Free/Open Source Software (FOSS), Open Content and Open Standards in the Asia-Pacific region. It is a network with a small secretariat based at the UNDP Regional Centre in Bangkok and three centres of excellence – IOSN ASEAN+3, IOSN PIC (Pacific Island Countries), and IOSN South Asia, based in Manila, Suva and Chennai respectively.

IOSN provides policy and technical advice on FOSS to governments, civil society and the private sector. It produces FOSS awareness and training materials and distributes them under open content licenses. It also organizes awareness raising, training, research and networking initiatives to assist countries in developing a pool of human resources skilled in the use and development of FOSS. IOSN works primarily through its web portal www.iosn.net that is collectively managed by the FOSS community. The web portal serves as a clearinghouse and a platform for knowledge sharing and collaborations.

www.iosn.net

Also available from UNDP Asia-Pacific Development Information Programme

e-Primers on Free/Open Source Software

- Free/Open Source Software – A General Introduction
- Free/Open Source Software – Education
- Free/Open Source Software – Government Policy
- Free/Open Source Software – Localization
- Free/Open Source Software – Open Standards

www.iosn.net

e-Primers for the Information Economy, Society and Polity

- The Information Age
- Legal and Regulatory Issues in the Information Economy
- Nets, Webs and the Information Infrastructure
- Information and Communication Technologies for Poverty Alleviation
- Internet Governance
- e-Commerce and e-Business
- e-Government
- ICT in Education
- Genes, Technology and Policy

www.apdip.net/elibrary

International
Open
Source
Network

An initiative of the UNDP Asia-Pacific Development Information Programme and supported by the International Development Research Centre, Canada

UNDP

with support from

IDRC CRDI

IOSN

c/o Asia-Pacific Development Information Programme
UNDP Regional Centre in Bangkok
3rd Floor, United Nations Service Building
Rajdamnern Nok Avenue
Bangkok 10200, Thailand
Tel: +66 2 288 1234; 288 2129

IOSN ASEAN+3

National Telehealth Center
University of the Philippines Manila
Taft Avenue, Manila, Philippines 1000
Tel: +63 2 525 6501

IOSN PIC (Pacific Island Countries)

The University of the South Pacific
Private Bag
Suva, Fiji
Tel: +67 9 323 1000

IOSN South Asia

Centre for Development of Advanced Computing
Block-II, 6/13, Park Avenue, Keshava Perumal Puram
Chennai-600 028, India
Tel: +91 44 2461 0880