E

WIPO
WORLD
INTELLECTUAL PROPERTY
ORGANIZATION

WIPO/CR/WK/GE/11/3
ORIGINAL: ENGLISH
DATE: NOVEMBER 8, 2011

Workshop on Using Copyright to Promote access to Information and Creative Content

Geneva, November 16, 2011

USING COPYRIGHT TO PROMOTE ACCESS TO INFORMATION AND CREATIVE CONTENT. SOFTWARE DEVELOPMENT PRACTICES (Part II)

prepared by Rishab Aiyer Ghosh[1]

[1] The views and opinions expressed in this Study are the sole responsibility of the author. The Study is not intended to reflect the views of the Member States or the WIPO.

TABLE OF CONTENTS

1. INTRODUCTION

Recommendation 19: To initiate discussions on how, within WIPO's mandate, to further facilitate access to knowledge and technology for developing countries and LDCs to foster creativity and innovation and to strengthen such existing activities within WIPO.
Recommendation 24: To request WIPO, within its mandate, to expand the scope of its activities aimed at bridging the digital divide, in accordance with the outcomes of the World Summit on the Information Society (WSIS) also taking into account the significance of the Digital Solidarity Fund (DSF).

Recommendation 27: Facilitating intellectual property related aspects of ICT for growth and development: Provide for, in an appropriate WIPO body, discussions focused on the importance of intellectual property related aspects of ICT, and its role in economic and cultural development, with specific attention focused on assisting Member States to identify practical intellectual property related strategies to use ICT for economic, social and cultural development.

WIPO Development Agenda, A/43/16

The WIPO General Assembly, in formulating the Development Agenda, recommended actions as cited above, to investigate how access to knowledge and technology for developing countries and LDCs could be facilitated. This Study aims to examine the practical strategies used in Member States to support economic, social and cultural development through the application of copyright regimes to software development practices. Drawing on numerous quantitative surveys of software use in developing countries, this Study focuses in particular on the economic aspects of software development under alternative models of copyright, i.e. open source software. The Study also examines specific cases of public policies, strategies and public institutional support of models for software development that facilitate wide access of software.

The treatment of software under copyright regimes has particularities which affect how public policies can address economic, social and cultural development issues. This Study therefore commences with a discussion of the treatment of software under IPR regimes; specific exceptions and limitations available or utilized; the alternative development model of open source software, which is in fact founded on copyright law.

An analysis of economic aspects of the open source software model follows, examining the incentives, costs and benefits from the perspective in particular of developing countries. Empirical evidence on economic and policy aspects of open source is then examined, drawing on numerous quantitative surveys and studies from Africa, Asia, Latin America, Europe and North America. This discussion is followed by an outline of legislative, fiscal and other measures that *could* be used to support software development, including a taxonomy of possible policies, and actually implemented policies across the world.

This is followed by a qualitative (and necessarily subjective) summary of selected case studies from different countries. A key factor in selecting the cases was their suitability for reproduction and transfer to other domains and regions, and the availability of public documentation and dissemination of information. In this regard, specific case studies were selected for initiatives that originated in certain countries and were actually reproduced in other parts of the world.

Finally, conclusions are drawn regarding the policies that should be considered by Member States in order to facilitate software development with optimal economic and social impact; and specific recommendations made for WIPO regarding its possible future role in this field.

2. IPR REGIMES FOR SOFTWARE: COPYRIGHT, OPEN SOURCE AND LIMITATIONS AND EXCEPTIONS

2.1. SOFTWARE COPYRIGHT

Software is covered by copyright law as a literary work[2]. Although software works are unusual in that they may also covered by patent law, the scope of this study is limited to software development practices with respect to copyright, the primary means of IPR protection applied to software works.

2.1.1. Legal Background

Internationally copyright law has historically been governed by the Berne Convention, agreed on since 1886, with various amendments up to 1979. The Berne Convention sets out a number of rights that are granted to the creators of literary or artistic works. The rights granted by the Convention include economic rights, specifically the right of the rights-holder to exclusively authorize the reproduction of works. Independently of the author's economic rights, and even after the transfer of the said rights, the Convention provides the author with "moral rights" - the right to claim authorship of the work and to object to distortion or mutilation of the work.

The Berne Convention did not specifically protect software authors, and the first legislative protection of software was in 1980 when the US amended its copyright law to include software[3]. In 1991 Council Directive 91/250/EEC made explicit the inclusion of software under European copyright law. The World Trade Organization TRIPS agreement in 1994 included software ("Computer programs") as subject matter to be protected as literary works, in Article 10. And the WIPO World Copyright Treaty (WCT) detailed rights applicable to software in 1996.

The legal framework for software copyright means, in summary, that a work of software cannot be used, modified, copied or distributed without the explicit permission of the software's authors. Although there are some limitations and exceptions of copyright that have been applied to software, in general, the legal framework means that the use of software is governed by the terms of a *license* from the rights-holders that determine how and whether the software can be accessed, used, modified, copied and distributed.

2.1.2. Policy initiatives

Public policy initiatives that aim to support economic and social development through the application of copyright to software development practices can take a number of forms: legislative actions affecting the scope and implementation of copyright; legislative and regulatory actions taking advantage of exceptions and limitations in copyright regimes as

---

2 In US law since 1980, followed by legislative changes around the world. See e.g. Mark A. Lemley, Peter S. Menell, Robert P. Merges, Pamela Samuelson, Brian W. Carver. 2011. *Software and Internet Law, Fourth Edition*. Aspen; Bridget Czarnota & Robert J. Hart, 1991. *Legal protection of computer programs in Europe: a guide to the EC directive.* London: Butterworths

3 Title 17, US Code, Sections 101 and 117

applied to software; and legislative, fiscal and other policy initiatives related only indirectly to copyright law.

While portions of this study touch on the first two forms, the primary focus of this study is on initiatives that work completely, in legal terms, within the standard copyright system. This is because the past nearly three decades has seen the rapid growth of software development practices that, originating in legal innovations in the application of copyright law, have evolved to a major economic methodology with spillover effects in non-software domains, cultural and societal practices. Free software, later also known as open source software, is a software development and licensing model that has been the primary alternative to traditional software development practices and a means of increasing software access through policy initiatives. In the business community, as later sections of this study show, the success of open source software has been large enough that it is part of the mainstream, and it calling it an alternative model is a misnomer.

Although the software development and economic practices in open source may differ from that used by proprietary software companies, when it comes to the *legal* aspect, open source does not rely on any exceptions from copyright law, and fits completely within the traditional copyright legislative framework. As this fact is key to the exploration of how public policy can increase access to software – new legislative approaches to copyright are not required – this Study provides below an explanation of how intellectual property rights and open source software development relate.

2.2. MECHANICS OF RIGHTS PROTECTION.

Open source software developers have become among the most economically productive online communities; however, there is sometimes a misconception that laws are ignored or that the community's efforts are "shared" as public domain and thus ignorant of IPR concepts. In fact, open source communities are among the most formalized in cyberspace, with the basis of their functioning guided by licenses under which their output is distributed, based on copyright law as a foundation.

The interaction between open source communities and rights is a complex interaction between the actors (developers and other community contributors), artefacts (code and documentation) and legal frameworks as they determine the scope of intellectual property rights.

To elaborate on this interaction, it is useful first to examine the way in which rights are treated by the current legal framework for copyright. Normally, once a work is created, it is exclusively appropriated by the creator, with a limited, temporary monopoly granted by the state. This monopoly provides the creator with the sole right to control access to the work; with copyright, the monopoly is over the reproduction of the work. It prevents follow-on creation by others without the permission of the creator. This monopoly is meant to reward the creator and provide an incentive for future creation.

With open source, this monopoly for the creator, providing rights to the created artefact, is not used as an incentive to create. As seen below, incentive structures in open source communities are more closely aligned to sharing of output rather than its appropriation. This introduces several complexities in the interpretation of who the creator is, and how (and by whom) rights are exercised.

2.3. RIGHTS CLAIMED BY OPEN SOURCE DEVELOPERS

It is essential to clarify that open source developers *do* claim and exercise rights over their creations, even if this is done through unconventional uses of the legal framework. Open source artefacts – software, documentation – are not public domain, in the legal sense of the term[4], though they may be public goods in the economic sense. Open source refers to software to which the "Four Freedoms" adhere (Stallman[5]): users have the freedom to use, freedom to study, freedom to modify and freedom to share this software.

While this includes works that are actually in the public domain[6], by default, software authors own their code. Under the Berne Convention, all copyrightable works are automatically covered by the copyright of the original creator at the moment of creation. No registration or notice – not even a copyright notice attached to the work – is required. Since software authors own their code, they are free to sell it, or indeed to "give it away". They must do this explicitly, and can impose conditions, which may perpetuate the "Four Freedoms".

Although the open source community has evolved its own implicit and explicit, informal rules and norms, the legal foundation of the open source community structure is in copyright law. Authors have the sole right to license their software to others, and software users must follow license terms – otherwise they are infringing authors' copyright.

While software authors can safely "give it away", this would literally be releasing software into the public domain and disclaiming all future rights to it. This is rare (and not even possible in some legal frameworks, e.g. in jurisdictions which provide for inalienable moral rights of the author). Instead, licenses for open source follow two broad models: permissive and reciprocal, and both involve the release to licensees of human-readable source code along in addition to machine-readable object code.

2.4. MODELS OF OPEN SOURCE LICENSING

The *permissive* licensing model is fairly close to public domain. It allows licensees broad rights to use, study, modify, distribute the software with few if any conditions. Most conditions relate to disclaimer of warranty issues. Examples of such licenses include the Berkeley BSD license, under which the popular operating system FreeBSD and its relations are distributed; the Apache license used for the market leader in web server software, Apache; and the MIT license used for the X Window system of graphical user interfaces under Unix-like operating systems. As the names of some these licenses indicate, they originated in universities and are often referred to as *academic* licenses.

The other licensing model, accounting for a majority of open source projects is *reciprocal*. 'Reciprocal' is used here to convey the notion that rights are being granted by the software authors, but *in return* (i.e., reciprocally), the recipients of the software must also grant similar rights if they redistribute the software. Quite different from the public domain, this model forms a "protected commons". Licensees have broad rights to use and study the software. If

---

4 With no claim of (copy)right, works in the public domain can be used in any way by any one; see Samuels, Edward, 1993. "The Public Domain In Copyright Law by Edward Samuels", *Journal of the Copyright Society* 41:137.

5 Stallman, Richard. "The Free Software Definition". Available at http://www.gnu.org/philosophy/free-sw.html. The definitive list of open source software licenses is maintained by the Open Source Initiative, following the Open Source Definition, see http://www.opensource.org "

6 E.g. software created by employees of the US Federal Government, which uniquely under US law is, like other intellectual works created by US Federal Government employees, in the public domain.

they distribute the software, they must provide recipients access to the source code (providing them the *freedom to study*). They must also provide recipients with the software under the same terms, allowing recipients the freedoms to further *use, modify or distribute* it. Licensees can only modify the software if the modified software is also distributed under the same terms. All recipients of such a derived work can, according to the original license, further modify it. This ensures reciprocity by forming a "protected commons" – authors are contributing their software into a commons with certain freedoms attached, and any further modifications must be made available with the same freedoms provided. This principle of reciprocal licensing can be described in lay terms as: "I am giving you certain rights over my software, and if you distribute this software, you must ensure that recipients receive the same rights from you as you did from me".

The best known reciprocal license is also the most widely used open source license, accounting for over 66% of open source software projects (Freshmeat 2005[7]), the GNU General Public License (GPL), with a further 6% distributed under the closely related Lesser GPL. The GPL is the license used by the Linux kernel and several other large software packages. Other widely used reciprocal licenses include the Mozilla Public license[8], used for the popular web browser Firefox; the Lesser GPL[9], used by Open Office, the main competitor to the Microsoft Office productivity suite; the European Union Public Licence created by the European Commission for the release as open source of publicly funded software[10].

2.5. CONDITIONS AND "RECIPROCITY"

Note that this "protected commons" created by reciprocal licenses is not formalized, and there is no obligation that licensees who modify software to make derived works "give back to the commons" in a formal sense, i.e. modified software does *not* need to be given away at no cost; nor does source code need to be published or provided to the original author. Indeed, such requirements would disqualify a license from being a free software (or open source) license. Reciprocal licenses such as the GPL require that *recipients of software* have the four freedoms; they do not require that the public at large have these freedoms.

The GPL, for instance, allows an author of a derived work to sell the work for 5 000 Euro a copy in only binary form (machine readable object code). However, all those who buy this software must, according to the GPL, be given the four freedoms. In particular, they must have the right to study the code, which is why the GPL requires that *recipients of object code* – in this example, the buyers – be provided with the source code at no significant extra charge. Similarly, the recipients have the right to modify and distribute the code with no further conditions; since they may distribute the code they received at no charge, if they so wish, or sell it for a lower price than they paid for it, charging high prices for the code alone, while allowed by open source licenses, is unsustainable under normal market conditions.

The reciprocal conditions imposed by open source licenses such as the GPL are unusual, though they have since been widely reproduced (including in Creative Commons licenses for non-software works such as art and text). Several commentators have raised questions as to the validity of the GPL's reciprocity requirements, going so far as to claim that reciprocal

---

7 http://freshmeat.net/stats/#license - 66% when accessed on July 17, 2006

8 http://www.mozilla.org/MPL/MPL-1.1.html

9 http://www.gnu.org/licenses/lgpl.html

10 http://www.osor.eu/eupl

conditions expropriate the intellectual property rights of the authors of derivative works[11]. However, a derivative work is a work built upon an original work, such as a modified version or extension of an original work of software. Authors of derivative works have no particular right to create them in the first place. Copyright law prevents anyone from modifying or distributing software without the explicit permission of the copyright holder – permission granted usually through a license. The copyright holder, choosing to grant the permission to modify, is free to set any conditions on the license.

For example, an open source license gives a person permission to sell copies of the licensor's software, which would be forbidden (without permission) under copyright law. It may place conditions on this permission. But a open source license normally cannot place conditions on your ability to copy parts of software, make a personal backup copy, or other activities allowed by copyright law under "fair use" or equivalent[12] terms.

Without following licensing conditions, users who distribute a work or make derived works are making unauthorized copies, thus infringing copyright. Indeed, in a rare court case concerning the distribution of modified versions of netfilter/iptables, a tool in Linux, a German appeals court ruled that even though the GPL itself may not entirely be valid in German contract law, it was the only thing that granted permission to the accused to distribute the software. Thus, the terms had to be obeyed, otherwise it was a simple case of copyright infringement.[13] Several incidences of violations of open source licensing terms have since been identified, mostly settled out of court[14].

2.6. RECIPROCITY AND COLLABORATION

Creators of a number of open source projects, with the aim of maximizing use, have chosen permissive rather than reciprocal licenses. Some of the early choices have been without much discussion, almost by default. The prototypical permissive license is the BSD license[15], used for the various versions of BSD Unix (the "Berkeley Software Distribution"[16]). This enormously influential systems software suite has, in great part due to its license, provided the underlying operating system for all Apple Macintosh computers since 2002 (and is the core of Apple's iOS mobile operating system, meaning that the iPhone and iPad run on open source software). Originally copyright of the Regents of the University of California, the BSD license was typical of the "academic" publication ethic.

Similarly brief and permissive (and academic in origin), the MIT License[17] or X license originated to distribute the X Window System[18], the graphical user interface (GUI) for Unix

---

11 "This viral aspect of the GPL poses a threat to the intellectual property of any organization making use of it", in Mundie, Craig. 2001. "Speech Transcript - Craig Mundie, The New York University Stern School of Business", May 3, Available at http://www.microsoft.com/presspass/exec/craig/05-03sharedsource.mspx; See also Evans, David S. and Reddy, Bernard J., 2003. "Government Preferences for Promoting Open-Source Software: A Solution in Search of a Problem", 9 *Mich. Telecomm. Tech. L. Rev*. 313. Available online at http://www.mttlr.org/volnine/evans.pdf; see also Jonathan Schwartz, CEO of Sun Microsystems, quoted in Profitt, Brian. 2005. "Editor's Note: With Friends Like These...", *Linux Today,* April 8, available online at http://www.linuxtoday.com/it_management/2005040802526OPBZ

12 E.g. in many European countries, the right to make a "private copy"

13 District Court of Munich I, Judgement of 19/05/2004 – file reference: 21 0 6123/04; English translation available at http://www.oii.ox.ac.uk/resources/feedback/OIIFB_GPL2_20040903.pdf

14 See http://gpl-violations.org

15 http://www.freebsd.org/copyright/freebsd-license.html - permissive licenses are often called "BSD-like"

16 Much information and history is available on Wikipedia: http://en.wikipedia.org/wiki/Berkeley_Software_Distribution

17 http://www.opensource.org/licenses/mit-license.php

18 http://en.wikipedia.org/wiki/X_Window_System

that originated in 1984 at MIT and is now the basis for most GUIs running on versions of Unix, Linux and BSD.

Both these licenses were implemented before the first version of the GPL, thus before the notion of reciprocal licensing became widely known. The GPL's legal innovation was truly remarkable, and the most significant permissive license that followed was probably the Apache license in 1995[19]. This was written for the Apache web server, an open source application written *not* by academics but by Internet professionals and website administrators. The GPL was already the dominant open source license and the discussion among the Apache developers, about whether or not to require reciprocity, is something many subsequent projects have faced, with varying degrees of argument. Apache chose to maximize its user base, and to encourage contributions to the commons through gentle social pressure[20] rather than legally binding restrictions. Indeed, Apache's user base was maximized – it became the most used web server within a year of its release, and has held a steady two-thirds of the total web server market since 2000.

Some of the scripting languages and content management systems - tools used (among other things) to make websites interactive – have also used permissive licenses. But the Linux kernel and the majority of open source software use reciprocal licenses. One reason is that reciprocal licenses are drafted to enforce reciprocity through "recursion" – typically, a derived work must be distributed under the *same* license. Thus, new software that reuses old GPL software – code reuse being one of the hallmarks of the open source software development model – must be licensed as GPL. For those not strongly opposed to reciprocal licensing, choosing the GPL is a fair trade for getting access to an ever huger codebase to reuse. Thus each reciprocal license is automatically designed to be dominant, and the most popular, or oldest, will by default dominate the entire license space.

However, one cannot say that most developers are against reciprocal licensing, or even neutral towards it. For rational actors, reciprocal licenses may be a better choice than a permissive license. Certainly, one feature of "giving your work away" that is hard to justify by a shortsighted rational actor is the threat of competitors benefiting from what you give away, or more generally, the threat of free-riding. This occurs less with reciprocal licenses, since competitors can benefit, but they cannot *exclusively appropriate* the benefits. They can "share", but not "steal". If they adapt or improve the work, they must in general return it to the commons, allowing the original creator to benefit from the improvements. Reciprocity ensures that development remains collaborative, and cannot be exclusively appropriated. The reciprocal licensing model allows the Linux operating system, for example, to have several thousand individual copyright holders, for each separate contribution made – something that would be quite impractical if individual licensing agreements had to be made. This is an example of how open source licensing lowers transaction costs for collaboration.

2.7. RECIPROCITY AND INCENTIVES

Reciprocity provides incentive for new contributors, including firms. 60% of developers think[21] the role of a license is "To prevent others from appropriating the software we've

---

19 The current version is 2.0, written in 2004 and available at http://www.apache.org/licenses/LICENSE-2.0

20 See e.g. Apache Software Foundation, 2006. "Frequent Questions about Apache Licensing". Available at http://www.apache.org/foundation/license-FAQ.html

21 out of 1540 respondents: http://www.stanford.edu/group/open source-us/stats/q7.html

created" (open source-US survey[22]), thus showing that they are not altogether (if at all) altruist and may frequently be choosing reciprocal licenses with the selfish motive of ensuring their access to future improvements.

The preference among developers for reciprocity is not limited to independent individuals. According to a survey of Italian firms that release open source software, firms prefer to use the GPL because "it allows to keep the code open and forbids competitors to turn it into proprietary." (Bonaccorsi & Rossi 2003[23]).

This has even been a concern for the public sector. For example, in a study conducted to examine the possibility of the European Commission releasing a software application it owns under an open source license, a key condition was that "the Commission requires protection against appropriation of application by third parties" (Dusollier, Laurent and Schmitz 2004[24]). The recommendation, based on this requirement, was to use a license with a reciprocity clause, i.e. a copyleft license such as the GPL.

2.8. RECIPROCITY AS A FRAMEWORK FOR DISCLOSURE

Patents, which are justified on the basis of promoting disclosure (and therefore follow-on innovation), are not really succeeding at that task, according to a number of surveys of innovators.

Arora et al (2003)[25] find that "patent disclosures appeared to have no measurable impact on information flows from other firms, and therefore no measurable effect on R&D productivity". Arundel (2001)[26] finds that "a consistent result in survey research on the use of patent databases is that they are among the least important external information sources available to firms". His analysis of 12445 firms' responses to the CIS survey results[27] shows that between 5% and 18% of small and medium-sized firms find patents to be a useful source of information[28].

In the case of software, surveys show (Arundel et al 2006[29]) that more firms think free software source code is an important source of new ideas (17%) than patent databases (5%). The opinion of individual innovators (engineers) is perhaps more relevant as

---

22 David, Paul, Waterman, Andrew and Arora, Seema, 2003. "FLOSS-US: The Free/Libre/Open Source Software Survey for 2003". *SIEPR/KNIIP Working Paper,* available at http://www.stanford.edu/group/open source-us/report/open source-US-Report.pdf

23 Bonaccorsi, A. and C. Rossi (2003). "Licensing Schemes in the Production and Distribution of Open Source Software: An Empirical Investigation". MIT Open Source working paper series. Available online at http://opensource.mit.edu/papers/bnaccorsirossilicense.pdf

24 Dusollier, S., Laurent, P., and Schmitz, P-E. 2004. *Open Source Licensing of software developed by The European Commission (applied to the CIRCA solution)*. European Commission DG ENTR. Available online at http://europa.eu.int/idabc/servlets/Doc?id=19296

25 Arora, A. et al., 2003. "R&D and the patent premium", Nat'l Bureau of Econ. Research, Working Paper No. 9431. p17. Available at http://www.nber.org/papers/w9431

26 Arundel, Anthony. "Patents in the Knowledge-Based Economy", *Beleidstudies Technologie Economie 67;*

27 Arundel A. (2000), "Patent – the Viagra of Innovation Policy?", *Internal Report to the Expert Group in the Project "Innovation Policy in a Knowledge-Based Economy"*, Maastricht, MERIT. Figure 4, page 15. Available online at http://www.ebusinessforum.gr/index.php?op=modload&modname=Downloads&pageid=320

28 the share is 34% for large firms, but even they find patents *less* useful than other sources of information, such as customers, suppliers, conferences and journals, trade fairs, and competitors.

29 Arundel, A., Bergstra, J., Feijoo, C., Ghosh, R.A., Glott, R., Hall, B., Klint, P., Martin, A., Thoma, G., and Torrisi, S. 2006. "Empirical Study of economic impact: Approach and preliminary findings". *European Commission,* part of the "Study of the effects of allowing patent claims for computer-implemented inventions", available online at http://www.merit.unu.edu/patentclaims/

questionnaires on patents sent to firms are likely to be answered by the legal department than by innovators. Far more innovators within firms[30] think software source code (41%) or journal publications (68%) are moderately or very important sources of new ideas, than patents (24%).

While we do not know how much of this software source code that is source of new ideas is licensed under reciprocal terms, these data show that open source software is succeeding in providing disclosure, while patents are less successful. This is certainly at least in part due to reciprocal licensing, which provides a legal requirement to disclose (much as patents are supposed to do). Without reciprocal licensing, disclosure would be only due to social, economic or other incentives, but not a *requirement,* and would presumably be reduced[31].

If a legal framework is required to promote disclosure and follow-on innovation, there is, therefore, some evidence to justify an argument that reciprocal open source licensing provides a more effective framework than the current patent regime. At any rate, open source licensing has come to form an innovative layer above copyright law to further access to software as well as facilitate wider participation in the process of software development itself.

3. OPEN SOURCE STRATEGIES FOR LOCAL DEVELOPMENT: ECONOMIC ASPECTS, INCENTIVES, COSTS AND BENEFITS

"Access [to ICTs] is not enough, it is the ability to create, to add value, that is important"
Felipe Gonzalez, former Spanish Prime Minister[32]

What former Spanish Prime Minister Felipe Gonzalez referred to as the ability to create and add value is particularly important for developing countries and other economically disadvantaged communities. Access alone limits them to the role of passive consumers in the knowledge economy; the ability to create transforms them into active participants. By lowering barriers to the transfer of knowledge, reducing transaction costs and enabling a protected commons, open source arrangements for software development has been shown to provide a training environment that enables this ability to create; it increases the earning capacity of community participants without any explicit investment in training and is perhaps a novel form of technology transfer.

The common feature described in the literature[33] for various examples of collaborative innovation shows that the most important enabling feature is *access.* Access is not required to knowledge alone, but to the tools and (legal) ability to replicate and improve upon knowledge. Thus it is not access to knowledge as passive consumers, which is often discussed and fitted well with the old model of R&D where producers were distinct from

---

30 Arundel et al 2006 (*supra* note) shows consolidated data for all respondents; figures included here are for individual innovators employed at private companies, i.e. excluding those employed at public organizations or research institutes.

31 Several firms embrace disclosure for other incentives, e.g. when they contribute to Apache software which has no reciprocity requirements. However, several firms try to evade the disclosure requirements of reciprocal licenses such as the GPL, when the GPL's legal requirement to disclose provides a useful mechanism. The court case referred to in *supra note* is one example, and the GPL Violations Project (http://gpl-violations.org/) contains many others.

32 Gonzalez. Speaking at Open Source World Conference in Málaga, Spain, 18/2/2004. From author's notes.

33 e.g. Benkler, Yochai, 2006. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale Press; Ghosh, Rishab Aiyer (ed.), 2005. *Code: Collaborative Ownership and the Digital Economy*. MIT Press; Ghosh, Rishab and Soete, Luc. 2006. "Information and Intellectual Property: The Global Challenges". *Industrial and Corporate Change*, Vol. 15, No. 6, pp. 919-935

consumers. In this model, developing countries are often treated as consumers who do not have the ability to innovate, perhaps due to the lack of technical skills, and must therefore passively consume products of developed countries (with subsidies, if required) or if they are more industrially advanced they may imitate production methods developed elsewhere. Apart from being patronising, this view does not fit with the new mode of technological progress for development, for two reasons.

First, empirical research has shown (Ghosh and Glott, 2005) that in the case of software, open collaboration provided by access to modifiable technology may not be problematic due to a lack of skills; rather, it leads to the development of technical, business and legal skills. Such skills are often better than those learnt in formal courses and proven participation in open source development may compensate for the lack of formal degrees. These results were supported by employers surveyed. This shows that while access to knowledge may build skills through passive absorption (e.g. through textbooks), access to technology in a form that can be shared and modified without entry barriers (as with open source software) can build advanced skills, compensate for the absence of formal training and generate increased employment.

Second, the premise of the new mode of technology development is that lowering entry barriers for the modification of technology reduces search costs, allowing participants in the market of producer-consumers to more efficiently allocating skills and other resources to needs for improvement. This leads to more efficient and perhaps faster technical innovation, with the entrepreneurial risks of innovation spread widely. Thus, providing access to technology need not be seen as charity or aid for developing countries, but as enlarging the resource base of potential innovators.

While access to knowledge as a passive process is politically framed within the language of development aid, access to technology as a way of providing the right and ability of participation is analogous to the arguments favouring free trade: developing countries can then be seen as providing a resource of potential innovators, rather than merely using existing innovations from the developed world.

This leads to the question of whether public policy should favour passive use of “black-box” software or active participants in the global ICT community. Being active requires being able to create – and choose with the least barriers the level of creativity. Clearly, the lower the entry barrier for creativity, the higher the potential that creativity that will occur. Developing countries need to avoid being locked out of skills and competencies. Skills development requires access to the ability to create, not only the access to software itself but to the process of software development, which as the following summary shows, is provided by the open source development model founded on the use of copyright licensing.

3.1. SKILLS DEVELOPMENT: INFORMAL APPRENTICESHIPS BENEFITTING EMPLOYERS

Open source, or free software as it was originally called, has become in recent years one of the most talked about phenomena in the information technology world. This is remarkable, not only for the usual reasons that open source has been around for many years as a volunteer driven success story before being discovered by big business and government — but also because it has largely developed quietly on its own without the headline coverage and glare of international attention that it now receives.

This in turn makes it more attractive to governments and policy makers. Countries around the world, regardless of wealth, are trying to bring citizens into the Information Society and

provide electronic access to government services. Many of them are considering open source software as a cost-effective means of doing so. Many more see an inherent injustice in requiring citizens and businesses to buy software from specific vendors in order to communicate with the government, and are looking at *open standards* – which allow different products from different producers, whether open source or proprietary software, to work together.

What is the special economic and social value of open source software, and how can it be harnessed? The Free/Libre/Open Source Software (FLOSS) study in 2002[34], a comprehensive study of several thousand developers and users worldwide, first showed that the most important reason for developers to participate in open source communities was to learn new skills — "for free". These skills are valuable, help developers get jobs and can help create and sustain small businesses. The skills referred to here are not those required to use open source software, but those learnt from participation in open source software communities. Such skills include programming, but also skills rarely taught in formal computer science courses, such as the basics of copyright law and licenses (a major topic of discussion in many open source software projects). Teamwork and team management are also learnt – after all, the team management is required to coordinate the smooth collaboration of 1500-plus people who rarely see each other can be more intensive and subtler than what is required to coordinate smaller teams employed in a single software company.

A large-scale follow-up study in 2005 for the European Commission under the FLOSSPOLS project found that developers *as well as employers* find that skills learnt by participation in the open source software community are so valuable that they may compensate for the lack of a formal degree. Large surveys for the European Commission under the FLOSSWorld and FLOSS include research projects in 2007-2010 – the first large surveys on FLOSS, with thousands of respondents, conducted across developing countries and LDCs in Africa, Asia, and Latin America – found similar results regarding the economic value of open source software for the development of local skills of use to employers.

Open source communities are like informal apprenticeships – but the apprentice/students and master/teachers contribute their own time “for free”, without any monetary compensation for the training process. Everyone can benefit equally from this training – any employer can hire someone informally “trained” through participation in the open source software developer community. However, not everyone invests equally in it. As many “teachers” may have been formally trained at university or at work, which is explicitly paid for, explicit costs are being borne for some proportion of community participants who have been formally trained.

In the larger perspective, this training system where all parts of society benefit from the products of the system, but only some explicitly pay for it, represents a subsidy – or technology transfer – from those who pay for formal training to those who do not (or cannot). Within countries, this represents a technology transfer from big companies who often formally pay for training to small and medium-sized enterprises (SMEs), who can less afford formal training expenses. Globally, this represents a technology transfer from the usually richer economies who can afford formal training, to the usually poorer ones who cannot.

---

[34] Ghosh R., Glott R., Krieger B., Robles G. (2002). *Free/Libre and Open Source Software: Survey and Study, **FLOSS**, Final Report*. European Commission / International Institute of Infonomics, University of Maastricht. See Survey of Developers. Available online at: http://www.flossproject.org/report/index.htm

3.2. BUILDING LOCAL ICT COMPETENCIES

Local skills development extends to the creation of new, local businesses, which are able to provide commercial support for and build upon open source software thanks to its low entry barriers, in a way that would not be possible with proprietary software where standard copyright licenses prevent transaction-free access by third parties. This effect is heightened by any public support of the open source software sector.

Facilitating local software development is especially important given the natural tendency of traditional global proprietary software vendors to ignore local needs especially in developing regions. As proprietary vendors are motivated by global profit-maximization strategies, local issues and user needs take a lower priority. So, for instance, a large multinational software company may not be interested in supporting Xhosa speakers in southern Africa. And since their software is proprietary, no local user or local business is in a position to add such support. Open source developers in Europe or North America may similarly be uninterested or unwilling to develop support for Xhosa speakers. However, making software available under open source licenses allows developers in southern Africa to learn from and adapt it to support Xhosa. As the description of this case later in this study shows, local development of software for local communities can then result in localization infrastructure that, while built in Africa, is later exported and reproduced in other parts of the world.

Such local adaptation supports the creation of new, local businesses, which are able to provide commercial support for and build upon open source software thanks to its low entry barriers, in a way that would not be possible with proprietary software. This effect is heightened by any public support of the open source software sector. For example, the take-up by the Extremadura Region in Spain of open source through its support for the LinEx project has led to an economic regeneration in a relatively poor region of the European Union (receiving, in April 2004, the award of the European Regional Innovation Award). This has not just allowed the implementation of activities for a lower price, but activities especially in education and training which were simply not possible with proprietary software; it has also led to the growth of a number of small businesses to provide commercial support, since with open source software there is no need for customers to approach one sole vendor for support — approaching local entrepreneurs is possible and an obvious choice.

For SMEs who do not already have extensive ICT use – and this applies to significant sectors and regions of the economy – evidence from MERIT's initial study of the impact on local firms of the ICT/open source policies of the regional government of Extremadura[35], Spain is instructive. There is a clear indication that while open source use may not in itself drive economic growth, the availability of open source drives ICT (not always open source) take-up among SMEs. A significant connection between ICT performance in firms and the role of open source was found. There was strong evidence that effective ICT performance together with the role of open source is what counts in terms of improving firm performance: above average performing firms with respect to ICT performance and open source support exhibit above average scores with regard to market share, cash flow and return on investment.

This performance seems driven by the importance given to innovation, and a close relationship was found between ICT use together with open source use and educated employees, and the degree of innovation. Thus, besides ICT importance in general (which is the most important indicator when compared to other firms with a lower ICT use), open

[35] Dunnewijk, Theo and Garcia, Abraham, 2005. The economic impact of ICT policies in Extremadura. FUNDECYT/Junta de Extremadura, Badajoz, Spain

source support seems to be part of the explanation for the actual ICT performance together with the level of education of the employees. The conclusion was that ICT performance matters and open source support and the level of educational attainment are equally important for its performance. In particular, a number of local small businesses have arisen to support and develop open source applications, sell hardware based on open source (in particular, Extremadura's version of Linux called gnu/LinEx). Some of these also develop new software, such as FacturLinex, a open source invoicing and billing system developed by a local micro-enterprise and used in many shops in Extremadura and increasingly elsewhere in Spain. In interviews with MERIT, small business customers in Extremadura have expressed a preference for using software which a small firm has developed (or helped to develop) as they feel they will get better support and personalised attention, whereas a large firm with a proprietary product may not be willing or able to attend to their specific needs. It should be noted that the Extremadura model has already been duplicated in other regions, especially in Spain, such as the much larger Andalucia, where about 400 000 desktops are running a localised version of the open source operating system GNU/Linux, which is also the standard platform – as with Extremadura – for libraries and digital inclusion centres. As pioneered by Extremadura, which used regional policy in support of open source to encourage local SMEs to provide IT services, Andalucia is also developing a regional policy to induce economic development through SME firms retaining a higher share of value added locally.

Of course, proprietary software also supports local businesses (excluding businesses who are *users*, who exist regardless of the type of software). What are the types of businesses that can be based upon proprietary software? Building new products and services above the platform is one, equally applicable to open source software – 100% of this value is local. Sales commissions are another, rarely possible with open source software, and of relatively low value. While 100% of the commissions may be locally retained, they represent a small proportion of the total value added, and every dollar of sales commission represents several dollars of imports. Finally, support, integration and customisation – this is where with proprietary software the local value added is limited by the proprietor's control of the software. Deep, high-value support requires deep, high-value access to the software, which only the proprietor has.

With open source software, the "deep support" that can be provided by "deep access" to the code available to all local businesses can generate enormous value, all of which is retained locally. No royalties or licences fees have to be paid.

Even for local businesses producing their own software, rather than only supporting other software, open source software is often a better value proposition: the licensing model allows providers to reuse software built by others without additional licensing or payment rather than build from scratch. The low transaction barriers means there it is possible to reuse a huge base of software written by others. Re-using (and modifying) allows the creation of much better end-user solutions for the same effort as compared to than creating completely new software, which local businesses are typically forced to do if they choose to develop software for sale under the proprietary software model. Put together, this provides better value for money for customers (who benefit from software representing a large base of cumulative development) and better profit margins for local service providers (who can focus on adding new features faster rather than replicating basic ones, allowing them to charge more for less work). Thus, access to software as well as participation to software creation is increased.

It must be emphasised here that increased open source software use can allow regional economies, and SMEs in particular, to locally retain a higher *share* of the added value. It is

clear that sales commissions related to proprietary software may lead to a higher *absolute* value retained locally, if proprietary software is much more widely used than open source software. A high added value in a small market can be less than valuable locally than low added value of a large market. Indeed, this makes open source potentially rather attractive, as it currently provides lower absolute added value locally than proprietary software, but provides a higher *share* of added value retained locally. This is because the market is currently dominated by proprietary software. Our analysis above suggests that if the share of open source software was increased relative to proprietary software – whether by market-driven demand, or by regional policies as described in this study – since the share of all value added that was retained locally would rise, the total value retained locally would also rise significantly. In any case, when a high share of proprietary software leads to a high absolute value added retained locally in the form of, say, sales commissions, this only indicates the even higher absolute value that is *not* retained by local firms.

4. LEGISLATIVE FINDINGS AND POLICY OPTIONS

4.1. LEGISLATIVE AND POLICY MEASURES TO SUPPORT WIDER ACCESS TO SOFTWARE

Unlike for some other creative works, initiatives to improve access to software do not seem to take advantages of any exceptions or limitations to rights. Although IPR laws for software could, in principle, be written to provide exceptions or limitations, those that are in place seem to serve specific, technical purposes: specifically, the limitation on copyright that allows for reverse engineering for the purposes of interoperability, in US case law and the EU Software Directive. There have been discussions in policy circles of using TRIPS Article 40 exceptions for software, but again this has been for the technical purpose of ensuring interoperability, and not increasing access to software.

In some situations, software copyright simply has not applied. E.g. in least developed countries temporarily exempt under TRIPS from enforcing software copyright, such as Cambodia (see case study), or jurisdictions with limited recognition or facing trade embargoes, such as Northern Cyprus – where proprietary software from US vendors simply cannot be sold, so in practice it must be widely copied ignoring copyright[36]. However, even in such situations, users, developers, industry, donors and policy makers have tended towards looking at open source software licensing as a forward-looking solution to providing and increasing access software. (In Cambodia, as the case study shows, it was to develop local-language software solutions for the first time; in Northern Cyprus, donor agencies including the European Commission and UNDP supported migration to open source software as a legitimate low-cost alternative to unauthorized copying of proprietary software for which copyright would be enforced after eventual unification.)

Policy initiatives examined here, therefore, exclusively relate to open source software as a means to increase access. Such initiatives can be classified as follows[37]:

---

36 A further, more esoteric case is when software is created by USA Federal Government employees; under US law, there is no IPR on such software, which is automatically in the public domain. There is one well-known case of software that was created in such a way: VistA, the health management software for the US Veterans Administration. It has since been further developed by foundations and the private sector, and a copyrighted version is distributed under an open source license (OpenVista).

37 This classification draws on: Wong, Kenneth. 2004. *UNDP-APDIP: FOSS Government Policy*. Elsevier. Available online at http://www.iosn.net/government/foss-government-primer/foss-govt-policy.pdf

- Mandating Free/Libre/Open Source Software (FLOSS)[38]: government requires the use of FLOSS for all or specific types of software
- Preferring FLOSS: government prefers the use of FLOSS for all or specific types of software
- Mandating Open Standards: this often has the effect of preferring FLOSS
- FLOSS Competency Centres: supporting initiatives that provide expertise and support for public authorities and others with questions about FLOSS
- A common strategy of many FLOSS policies is the creation of a FLOSS competency/research/compatibility
- Awareness raising: the most widely recommended and successful strategy as shown from a number of empirical surveys, simply raising awareness of FLOSS has the tendency to increase its use and development; such awareness raising is typically in the form of promoting or aggregating news, conducting case studies of best practices, etc. See the OSOR case study.
- Credit/Financial Assistance

A number of countries have had some success in implementing some or several of these policies and initiatives. The Brazilian Government has managed to foster the development of open source software in all areas of its ecosystem – education, public administration, health, industry In Latin America, Brazil stands out from the rest of the countries in the region due to the greater extent to which it has adopted and developed FLOSS, with levels comparable to countries such as India and China, due to the publication of regulations, mass migrations in public sector agencies and companies, FLOSS product development (goods and services) at the public universities and the creation of a collaborative portal for Community players. The European Union and certain EU member states have also taken several policy initiatives (see the OSOR case study) and are helped by having the largest number of individual open source software developers world-wide.

Countries with a higher level of FLOSS development and adoption, such as the United States (where however, the private sector leads by far in FLOSS initiatives), Australia, Germany, France, Spain, each demonstrate high levels of development in all parts of the ecosystem. The open source software development model is a globalizing model in which players use the Internet to take part in projects in a cooperative environment, regardless of the nationality of the player or the project, and there are rarely differences between geographical areas, either in terms of the workings of the communities or the associated business models. When initiatives do take off, therefore, they quickly lead to global links – see the KhmerOS Cambodia and Sahana Sri Lanka case studies.

The Center for Strategic and International Studies has, for the past few years, compiled a list of public initiatives taken by national, regional and local authorities worldwide[39]. A quantitative summary from the 2010 list is provided below.

Table: Regional distribution of Open Source Initiatives

Region	Approved initiatives				Proposed initiatives	Failed initiatives
	R&D	Advisory	Preference	Mandatory		

[38]. Free/Libre/Open Source Software is generally referred to in this document by the acronym FLOSS, a term used in a number of studies and policy documents in Europe, Africa and Latin America.

[39] Lewis, James A. 2010. *Government Open Source Policies.* http://csis.org/publication/government-open-source-policies

Europe	45	37	36	8	27	10
Asia	19	16	22	2	20	2
Latin America	8	6	12	31	15	11
North America	5	8	2	1	11	10
Africa	3	1	4	8	1	0
Middle East	1	2	2	0	2	0

Note: R&D initiatives are non-policy initiatives supported by public authorities or legislatures. Other initiatives are policy initiatives, relating to development, software procurement regulation, or release of publicly funded software under open source licenses.

4.2. FISCAL MEASURES

FLOSS software development may not be a charitable activity, although a majority of contributors remain independent individual volunteers[40]. However, when the software is released to the public, it is a charitable donation and treating it as such for tax purposes may be a simple and effective support mechanism. It should be noted that IPR donations are commonly used for tax deductions by firms especially in high technology sectors in the US. There has been considerable controversy resulting in a general investigation by the US Internal Revenue Service on the somewhat arbitrary valuations placed by firms on such donations, particularly on donations of patents to universities[41].

However, with FLOSS software, a simple lower bound valuation could be the time spent on development. While there are means of evaluating this based on the size of the codebase[2], which could be used as a control on time claims, these "donations" could also be valued on the basis of actual time spent as documented by timesheets.

It should be noted that the logic of equitable treatment for in-kind donations of FLOSS applies also to other non-software goods that are donated under such "information commons" schemes, such as music, text, scientific and other creative works distributed under (several, but not all) Creative Commons licences. A control for valuation may be somewhat more difficult for other artefacts where, unlike for software, substitution cost estimation metrics do not exist – but auditable time input at the opportunity cost of the donor's time can always provide a lower bound for the value of the donation.

It should be noted that there is no specific policy in place in member states for tax treatment of open source contributions that the author is aware of. The proposal above was included in a report published by the European Commission (2007) which noted in detail how it was consistent at least with US tax law.

5. SUPPORTING SOFTWARE DEVELOPMENT: SUMMARIZED CASE STUDIES

[40] See Figure 28, "Distribution of code output by individuals, firms, universities", in European Commission 2007.

[41] See e.g. Feder, Barnaby J., 2002. "Patent Donations Are Novel Corporate Gift", *New York Times,* November 17 (Finance News). Available at http://www.nytimes.com/ref/open/finance/17PATE-OPEN.html

Previous sections have examined economic aspects and survey data relating to open source software and economic development, and legislative and policy initiatives. In this section, a few cases of specific initiatives have been examined in more detail. They show how open source models for copyright have allowed public initiatives to rapidly develop access and deploy software systems with significant impact. Initiatives examined have sometimes been originated by government, but are often originated by civil society or industry and later supported by public organizations – underscoring the flexibility of open source licensing, which allows users and developers to bypass the transaction costs and times typical of traditional copyright exploitation models. The initiatives examined here have been selected specifically for highlighting the role of local software development, and exportability to other regions (see the table below).

Thus, the following case studies are presented here.

1. Sahana: the award-winning disaster management system created in Sri Lanka as a response to the 2004 tsunami. Deployed in Sri Lanka by the government's Center of National Operations (CNO), it was later supported by a number of public and private agencies and deployed with further development around the world, including in Indonesia during the 2006 earthquake, Peru in the 2007 earthquake, and Haiti during the 2010 earthquake.

2. Ushahidi: a crisis mapping, data collection and visualization system created in Kenya in the violent aftermath of the disputed 2007 presidential election, the Ushahidi system has been used to monitor elections in Mexico and India, deployed shortly after the 2010 earthquake in Haiti, used to monitor the effects of the 2011 earthquakes in Christchurch, New Zealand, and Japan.

3. KhmerOS: a software localization effort in Cambodia that built upon the South African *translate.org.za* multi-lingual localization system, and was then "exported" to Bangladesh

4. IT@Schools Kerala: an initiative of the regional government of Kerala, India, to use open source software in all state schools, that is similar to a number of initiatives elsewhere in the world

5. Open Source Observatory and Repository: a European Union project, providing a development environment and repository of open source software for public administrations across Europe and an Observatory of case studies and news to build a community of practitioners, which has drawn from and become a model for other similar initiatives.

6. Softwarepublico, a Brazilian public software portal initiated by the Government.

Case	***Origin***	***Funding***	***Key stakeholders***	***"Export"***
Sahana	Sri Lanka	Volunteers; Industry; SIDA	FLOSS community & industry; emergency response / aid agencies	Indonesia, Peru, Haiti

Ushahidi	Kenya	Volunteers; Universities; UN OCHA	FLOSS community, civil society	Haiti, Chile, New Zealand, Japan, Libya
KhmerOS	Cambodia	Local NGOs; UNDP, UNESCO, IDRC, AECID, InWent, Internet Society, Government	FLOSS community; development agencies; government	South Africa ("import"), Bangladesh, Bhutan
Kerala IT@Schools	India	Government	School teachers, FLOSS community, government	Spain ("import")
OSOR	Europe	European Commission	Public administration, contractors, developer community	EU Member states; "Parallels" in Brazil etc.
Softwarepublico	Brazil	Government	Public administration, contractors, developer community	"Parallels" in EU, elsewhere

Note: Funding agencies listed are: SIDA (Swedish International Development Agency); UN OCHA (Office for the Coordination of Humanitarian Affairs); InWent (now part of Deutsche Gesellschaft für Internationale Zusammenarbeit, the German aid agency); IDRC (Canada's International Development Research Centre); AECID (Spanish Agency for International Cooperation for Development). The Export column lists "imports": previous implementations upon which the described cases drew; and "parallels": cases similar to but not directly following the described case.

5.1. SAHANA: DISASTER MANAGEMENT IN SRI LANKA, PERU AND HAITI

Case summary	
Geography	Sri Lanka
IPR Issues	Open source licensed software development: the project involved the use of existing open source software and the adaptation and development of software released under open source licenses
Stakeholder incentives	Volunteers and local software industry responding initially to the catastrophic 2004 Tsunami and the lack of software tools to help emergency responders; emergency responders and aid agencies incentive to participate is the availability and development of unique software tools.
Sustainability	Economically sustainable through funding from donor agencies (emergency response) globally; commercial sustainability through furthering projects and brand image marketing for Sri Lankan software, and training and participation for local software developers and industry.

Impact	Widely recognized as the best and essential software tool set for emergency response to catastrophes, Sahana has been used around the world. It has had a major impact in terms of supporting emergency response and recovery of economies after natural catastrophes, but also in terms of the use of and access to Open Source software and Sri Lankan software developers' participation in the global software developer community.
Transferability	The software developed was adapted and used in various settings – indeed, in several of the major developing-country catastrophic natural events since 2005, including the earthquakes in Indonesia (2006), Peru (2007) and Haiti (2010), so it is clearly transferable. The process of development itself was special, though not unique; a number of other regions have seen the development of local software in response to local conditions, which once released as Open Source have received worldwide adoption. E.g. Ushahidi in Kenya, or GNU Health in Argentina.
Public policy implications	Sahana and Ushahidi are examples of civil society rapidly responding through the use and development of open source software to specific unmet local needs that turn out to be global and more broadly in demand, and develop local skills. Public authorities can support or even lead such initiatives, working with civil society to rapidly develop a local response and software developer community. If managed with local business foundations, such as with Sahana, this can also result in developing a global reputation for locally developed skills.

5.1.1. Tsunami

Sunday, December 26th, 2004: A devastating Tsunami hits Indonesia, Sri Lanka and many other Asian countries. In the first week of the tsunami in Sri Lanka, 1 million people (5% of the population) was rendered homeless, two-thirds of Sri Lanka's coast was damaged and nearly 40,000 people died.

Tuesday, December 28th, 2004: Many different organizations in Sri Lanka start efforts to write various bits of software to help manage the disaster. (This process also took place in other affected countries, including India, Indonesia and Thailand.)

Wednesday, December 29th, 2004: Software developers get together at the ICT Agency in Narahenpita, Sri Lanka to discuss ways of putting the software all together to make it easier to manage the situation. Sanjiva Weerawarana, Founder & Director of the Lanka Software Foundation (LSF, an industry body which supports Open Source software) called the US Federal Emergency Management Agency (FEMA)'s CIOs office and asked for whatever software they had, but was told that “FEMA had no software that could help; they only had software that was used to cut checks to people after hurricanes”.

In the 3-4 weeks that followed, many individuals, universities and software companies and Sri Lanka Telecom contributed to what became known as Sahana. While most contributors to the initial effort were from Sri Lanka, international communities of Open Source developers were also involved. Part of the initial development was done on computers that IBM donated within a week or so of the tsunami. The joint effort was coordinated by the LSF. Software was developed and went into production within a week. After about 3 months the initial phase of software development and deployment completed.

In the meantime, it became clear that there was a gap in the world of disaster management software. The state of the art that the UN team that came to Sri Lanka with was based on extremely outdated proprietary software. Existing solutions were not easily deployable or scalable and, most importantly, didn't embrace the Web. The tsunami provided a unique opportunity to look at disaster management in the modern world: despite the destruction, mobile phone and Internet networks were intact (or could be rapidly re-enabled for emergency use with portable transmission). Clearly, there was a huge need for modern software that could live in this world and help first responders and follow-up recovery be more effective at responding and managing a disaster.

"We were not going to let Sahana die; we decided we are going to make it into something the world can reuse readily", said Weerawarana. In 2005 Swedish aid agency SIDA approved a proposal to fund Sahana phase II (for $85,000). The justification for the development of disaster management software under open source licenses, made in the Sahana proposal to SIDA, bears quoting in full:

> *"Very few countries and organizations today can afford to invest a lot of resources in disaster management when there is no disaster present. While this is obviously true of poor, developing nations, it is also true of richer, developed countries as well because there are always higher priority items that need the funding. Worse yet, even if there are some national scale systems that may get deployed, it is very unlikely that regional and local level systems will ever get deployed if they cost any significant amount of resources.*
>
> *Because no one is willing to pay for the software, no one is willing to build it either. This is what we see in the world today – while disaster management software is critically needed, there is no complete commercial or non-commercial software solution that is widely available. Going the open source way can address both these concerns. Using the open source development model, it is possible to develop this software at a much reduced cost compared to pure commercial development models. This is true because while commercial entities are not willing to invest into these systems, there are hundreds and thousands of well-meaning IT professionals who are very happy to donate a few hours of effort to helping build such systems. We are already seeing this with the nascent Sahana project. Thus if there was a small team which was driving such a project, then it is possible to get a lot of assistance from the global IT community to make those systems truly exceptional.*
>
> *Going with open source approaches can also greatly reduce the deployment cost of this software in peace (i.e., non-disaster) times. The Sahana system, for example, can be deployed on any PC with just a Linux LiveCD (that is, a CD from which the entire system can be booted up and brought on-line). Thus, not only is it possible to run this on commodity, inexpensive hardware, it is in fact possible to not even have dedicated hardware around – just take any office PC and make that the "disaster management center"! In fact, that is how Sahana was first deployed in Sri Lanka – on a borrowed PC. (Later it switched to running on a borrowed server as the capacity requirements increased.)*
>
> *Thus, open source is the natural way to providing disaster management solutions."*

SIDA funding for Sahana was followed by additional grants from donor agencies and industry (both local as well as international, including IBM and Google).

Sahana was restructured with its own Board – members are all volunteers – with LSF remaining the underlying legal authority for the activities that the Sahana Board governs.

5.1.2. Transferability

Sahana has been deployed and adapted all over the world. Some examples have been listed below.

The 2007 Peru earthquake measuring 8.0 on the moment magnitude scale that hit the central coast of Peru on Wednesday August 15, 2007 and lasted for about three minutes. The epicenter was located at 150 kilometers south-southeast of Lima at a depth of 39 kilometers. 50% of the population was left homeless with over 500 deaths reported.[42] IBM Peru lead the Sahana deployment with the support of Lanka Software Foundation and Sahana community of Sri Lanka. The system was localized into Spanish. The project was coordinated by the Prime-Minister's office in Peru, with the objective of tracking relief items and co-ordinate relief efforts among personnel.

The 2008 Sichuan earthquake on May 12, 2008 in Sichuan province of China killed at least 68,000, injuring 374,176. The earthquake left about 4.8 million people homeless. Set up at the request of Chengdu Municipal Government and was deployed as a collaborative effort by IBM (CSR), Lanka Software Foundation of Sri Lanka (LSF), Sahana-community & the Trinity College, Sahana deployment in Chengdu was used to register shelters, track affected persons and manage relief personnel and supplies.

Deployed in 2007 as a measure of emergency preparedness, Sahana Disaster Management system is currently in function at the Office of Emergency Management (OEM) of New York City Council for New York Coastal Storm Planning. The Sahana system is capable of coordinating a mass evacuation of 6 million people in the New York City area in the case of a hurricane, and is continually being updated to accommodate the city's changing population. It currently tracks 26,000 relief workers, volunteer staff and evacuees in over 500 shelters. This project was carried out as a collaborative effort between the IBM Crisis Response Team (IBM CRT), IT Crisis of USA and Lanka Software Foundation of Sri Lanka (LSF).

In the afternoon of 12 January 2010, a 7.0 magnitude earthquake struck the poverty-stricken Caribbean nation of Haiti. The impact of the earthquake, occurring just south of the densely populated capital city of Port-au-Prince, was devastating as scores of multi-storied concrete structures in the capital and surrounding municipalities collapsed, killing tens of thousands instantly, injuring and trapping thousands of others beneath the rubble.[43] The Sahana Software Foundation and the Sahana community responded immediately, with a hosted instance of Sahana on a public website that served to fill gaps in the information management requirements of the massive relief operation. Other organizations deployed adaptations of Sahana tools – e.g. the National Library of Medicine (NLM), the world's largest medical library and an arm of the US National Institutes of Health (NIH), released a Sahana-based "Lost Person Finder", called "Haiti Earthquake Person Locator".

5.1.3. Impact in Sri Lanka

The immediate impact of the Sahana initiative was of course mostly felt outside the field of software – in the recovery from the 2004 tsunami. However, there has also been a clear

[42] See http://respere.org/deployments

[43] Source: Chamindra de Silva and Mark Prustalis, 2010. "The Sahana Free and Open Source Disaster Management System in Haiti" in *ICT for Disaster Risk Reduction Case Study 2,* published by UN-APCICT/ESCAP, May 2010.

impact in terms of Sri Lanka's role in software development. Open source software turned out, with Sahana as the prime example, as a great enabler for Sri Lanka to enhance its position in the global software ecosystem. Being a source of mission-critical software that has literally saved thousands of lives around the world provides a credibility that would be hard to earn. A key factor in this has been the open source license, which allowed for rapid deployment and adaptation of the software to local needs – without which the usefulness of the software, however technically capable, would have been severely limited.

Beyond Sahana, the key aspect of the LSF strategy has been "*to create a platform on which Sri Lanka can build",* according to Weerawarana. He draws a parallel to another sector for which Sri Lanka is well known – tea: "*Sri Lanka has a few companies which are now global consumer brands in tea. That was only possible because of the brand Ceylon Tea. Having that brand enabled differentiation and enabled our companies to leverage that to compete for consumer recognition and adoption."*

LSF's strategy is to create a group of people who are global contributors to FLOSS to such an extent that the world recognizes Sri Lanka – which compared to its much larger neighbour, India, has a small software industry – "*as a powerhouse of open source development*". This strategy has seen some success. For the first few years the global Summer of Code contest run by Google, University of Moratuwa Sri Lanka was the winner of grants. In the Apache software project – a global open source software application that powers 70% of the world's websites – there are more Sri Lankan software contributors than from any other country outside the US and a few western EU nations. Sri Lankan produced software, distributed under open source licenses, has been globally adopted, riding on the reputation of Sahana and contributions to Apache projects. "W*e have demonstrated what is possible if you take the best people in a poor developing country and give them the right tools, environment and opportunity to compete in the global stage. [The open source software model's] beauty is that it allows anyone to compete globally - it is not necessary to be in San Jose or Boston or London to compete!*", concludes Weerawarana[44].

5.2. USHAHIDI: MAPPING AND VISUALIZATION IN KENYA AND CHILE

Case summary	
Geography	Kenya
IPR Issues	Open source licensed software development; open content licensing. The project involved the use of existing open source software and the adaptation and development of software released under open source licenses; a major part of the use of the project related to geographical mapping of data points submitted by large numbers of individual volunteers.
Stakeholder incentives	Volunteers responding initially to the violence around the 2008 Kenyan elections; citizens and civil society; donor agencies and various public agencies.
Sustainability	Economically sustainable through funding from donor agencies globally; possible commercial sustainability through commercial applications of "crowdsourced" mapping technology
Impact	Widely recognized as an effective, rapid solution to collecting, organizing and visualizing geographic data, Ushahidi has been used around the

[44] Correspondence with the author.

	world. It has had a major impact in terms of supporting emergency response and recovery of economies after various crises, but also for tasks such as monitoring elections. It is a key example of appropriate mobile computing, with its dependence on and exploitation of mobile networks
Transferability	The software developed was adapted and used in various settings – to monitor elections in India, Mexico, Lebanon and Afghanistan; track unrest in the DR Congo; monitor medicine stocks in Zambia; map events and activities in Haiti, Chile, New Zealand and Japan after earthquakes and Libya after the violent events recently. So it is clearly transferable. The process of development itself was special, similar to Sahana. Ushahidi's model of locational input means that each time it is used, a community of participants and contributors is being built, demonstrating further its transferability.
Public policy implications	Sahana and Ushahidi are examples of civil society rapidly responding through the use and development of open source software to specific unmet local needs that turn out to be global and more broadly in demand, and develop local skills. Public authorities can support or even lead such initiatives, working with civil society to rapidly develop a local response and software developer community.

Following the violence in Kenya after the 2008 elections, it was apparent that a method to track events - being able to see where disturbances, crimes and other events were happening – was an important way to coordinate information from news sources as well as local people. The volunteer team behind Ushahidi rapidly developed a tool for Kenyans to report and map incidents of violence that they saw via SMS, email or the web. Within a week Ushahidi had gone to live deployment. The team behind Ushahidi became an organization that created a free and open source mapping and content management system which can be used by organizations worldwide in similar crisis-related situations. The main goal of the organization is to create a system that facilitates early warning systems and helps in data visualization for response and recovery.

Erik Hershman, director of operations at Ushahidi says[45], 'We take the stance that you go for the lowest common denominator, which is the SMS enabled mobile phone. So you take your Nokia 1100 and you say, "If we can make the technology work on this that's useful for people both on incoming messages and outgoing messages then we have something that's valuable and let's see what people do with it." The first iteration of that was in Kenya during the post election ballots. We quickly created a website. It was a mash-up of maps and incoming mobile phones messages that we called Ushahidi, which means testimony in Swahili, then what we did was get funding to build a global version of this.'

Following the initial deployment, Ushahidi received support from a number of donor agencies and foundations, especially for deployments in different regions. One such major deployment was in Haiti. As Zook et al[46] write, "When the magnitude 7.0 earthquake struck Haiti on January 12, 2010, there was an immediate need for maps. Emergency responders

[45] Source: UK Design Council, 2010. "Case study: Ushahidi". Available online at: http://www.designcouncil.org.uk/our-work/challenges/security/design-out-crime/case-studies1/ushahidi/

[46] Matthew Zook, Mark Graham, Taylor Shelton & Sean Gorman, 2010. "Volunteered Geographic Information and Crowdsourcing Disaster Relief: A Case Study of the Haitian Earthquake". *World Medical & Health Policy*. Vol. 2: Iss. 2, Article 2 (2010). Available online at: http://www.psocommons.org/wmhp/vol2/iss2/art2/

had to know where the people most in need were located and how to get assistance and relief to them. Large parts of Haiti and its capital, Port-au-Prince, lacked adequate coverage in the standard web mapping services […] that people in most of the developed world have grown accustomed to using. As one of the world's poorest countries, Haiti had simply not provided the kind of demand for online mapping that drove its expansion elsewhere. Post-earthquake, the demand for spatial information and online maps increased tremendously and, given the urgency of relief operations, the ability to crowdsource the data collection process became particularly important." Ushahidi was used to allow volunteers across Haiti to notify the system of events and geographic markers from different locations around Haiti, and visualize and monitor the results. Ushahidi's ability to receive input by text message (SMS) meant that in Haiti – as in Kenya and elsewhere – the simplest of mobile phones could be used to provide geographically marked, accurately time-stamped reports.

"Crowdsourcing", or using the collective power of large numbers of possibly anonymous individuals, has become an increasingly well known method of solving problems ever since the popularization and explosive growth of Wikipedia. It is a good illustration of the power of open source licensing to promote software access that one of the most effective, widely used and innovative applications of crowdsourcing – with real crowds of ordinary people using the simplest mobile phones – was developed not in Silicon Valley but in Nairobi.

5.3. KHMEROS: LOCALIZATION AND SOFTWARE TRAINING IN CAMBODIA

Case summary	
Geography	Cambodia
IPR Issues	Open source licensed software development: the project involved the use of existing open source software and the adaptation and development of software released under open source licenses. As a background - special LDC status under TRIPS allowed Cambodia to not enforce software copyright, removing one incentive to use Open Source Software (the zero license fee) as the effective license fee for proprietary software was zero.
Stakeholder incentives	Primary incentive for stakeholders (NGO, government and Development Aid Agencies) has been the adaptation of software to the local Khmer language, for which open source software was the most appropriate and cost-effective; a further incentive has been to spread the knowledge built up with other countries in a similar situation (Bhutan, Bangladesh)
Sustainability	The economic sustainability is two-fold; operational sustainability for new development of software is provided for through development funding (i.e. non-commercial). However, the output, software distributed under open source licenses, is by definition a sustainable, widely used, essential end-product regardless of funding for future developments.
Impact	From the initial impact – allowing Cambodians to use computers in their own language – to the continuing effect of training local software developers and enabling computer use in other countries, the impact has been high.
Transferability	As with most open source software localization efforts, this drew on previous knowledge and cases – specifically, translate.org.za, an effort to provide computer access in local South African languages (under open source licenses) funded by the Shuttleworth Foundation. The KhmerOS project itself has replicated parts of its activities in Bangladesh and

	Bhutan, demonstrating transferability.
Public policy implications	For many countries with local languages, whose users are disadvantaged in accessing ICTs, FLOSS provides a way to make computers accessible in languages previously unsupported. Localization initiatives improve access and revitalize vernacular communications, and provide local software development skills

5.3.1. Computing in Khmer

From 2004 to 2010, the KhmerOS / Open Schools Program has changed the map of Information and Communication Technology (ICT) in Cambodia, making access to technology widely available to citizens by the simple means of translating free software to Khmer (Cambodian) language, providing training, and supporting the government on policy-making and planning, to ensure that these computer programs are used.

This has led to a strategic advantage for ICT in Cambodia, by ensuring that software is available to people in their own language, and that all new high school graduates are familiar and comfortable with Open Source Software applications. This new situation facilitates penetration of Open Source Software in both government and the private sector, reducing the financial needs of all of them, and potentially ensuring faster deployment of ICT around the country.

The use of local language in computers enables access to information and communication tools for the 98% of the population that does not have sufficient knowledge of a foreign language to use computers that are not in Khmer. It allows teaching of ICT in schools, as the base for developing professional skills, while facilitating automation of government offices and SMEs, effectively reducing the digital gap. Widespread use of ICT in the local language eliminates an important barrier to economic development.

KhmerOS started in 2004 as a technical NGO-based project to localize and adapt to Khmer culture Free/Libre/Open Source (FLOSS) computer applications, producing also documentation and training materials for this software, as well as new fonts and keyboards that supported the standardization of the use of Khmer in computers.

After working for two years with other government agencies, in 2007 the Open Schools program started as a joint initiative between the Cambodian Ministry of Education Youth and Sport and the Open Institute (the NGO that houses the KhmerOS project). While using the software that had been developed, the goals of this new initiative were more centered on using ICT to improve the quality of Education and on offering ICT-based professional skills to high school students.

From 2007 to 2009 the Open Schools program has developed a five-year Master Plan for ICT in Education, created curricula for students and for teachers, as well as the necessary textbooks, and trained ICT teachers in all the schools in the country that have computers for education.

The participation and experience of the UNESCO Asia and Pacific Regional Bureau for Education, as well as its Phnom Penh office has been crucial to develop the ICT policy and to turn it into actual plans accepted by the Ministry of Education, Youth and Sport.

5.3.2. ICT context in Cambodia

In 2004 Cambodia was in the middle of transforming itself from a market with few computer users and some small computer shops into a country where the use of computers was starting to become common. New ISPs started to take a part of the Internet connection market, which did not grow as fast as expected, due to the high price of government-controlled connectivity. Most of the software available and used was proprietary, mostly unauthorized copies easily acquired in markets for a few dollars, and available in foreign languages (English).

The government was nevertheless working on ICT policy, trying to understand and unblock factors that might delay economic development for lack of access to ICT. In 2003 the National ICT Development Authority of the Royal government of Cambodia (NiDA) started to develop a National ICT policy with the support of the UNDP Asia-Pacific Development Information Programme (UNDP-APDIP). In 2004 a first draft of this policy was being publicized; it included simple provisions for the use of Open Source Software but nothing about localization or the use of Khmer language in ICT.

The lack of consideration for the national language was a product of lack of awareness on the advantages of the using local language software, as all those working on the policy and all computer specialists spoke English or other foreign languages well. As in some other developing countries, it was assumed that people who could afford computers would be able to (and intend to) use them in foreign languages such as English. Moreover, there was no short-term likelihood of Khmer support being available.

The International Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS) had granted Cambodia, as a Least Developed Country, a special moratorium until 2013 under which it did not have to have or enforce any anti-piracy laws. Least developed countries (LDCs) were accorded special and differential treatment pursuant to Article 65.5 and 66.1 of the TRIPS Agreement. They were not required to implement the TRIPS Agreement (except for national and most-favored-nation treatment) until January 1, 2006, and they were not prevented from reducing their level of TRIPS consistency prior to that date. The date for TRIPS compliance was subsequently extended until July 1, 2013.

The economical situation of Cambodia, and the uncontrolled use of proprietary software used without paying for licenses made Cambodia an uninteresting country for proprietary software companies, who did not see how to make a profit in Cambodia in the short run. They did not consider interesting investing in either preparing the software for the use of Khmer script, or translating it and adapting it to Khmer. No software in Khmer language was available in 2004.

Also, given again the economical situation, in which the cost of a computer (hardware only) was higher than the average yearly income for a Cambodian, duplicating this cost by paying for software licenses would have put computers even further away from the economic possibilities of most Cambodians. As only computers in English could be used, computers remained accessible only to the elite who knew or could learn English well.

5.3.3. Free and Open Source Software in Cambodia

In 2004 there was very little awareness of the existence of Free/Libre/Open Source Software (FLOSS) in Cambodia. Training institutions teaching the use of computers used proprietary software applications in English. UNDP-APDIP was a strong advocate of the use of FLOSS, and had at least participated in introducing the use of FLOSS in the draft ICT policy. The Government of Japan, through CCIC (Center for International Cooperation for Computerization), was organizing meetings all over Asia promoting the use of FLOSS in

governments. Officials of the Cambodian National ICT Development Authority (NIDA) had been attending these meetings since 2003. Cambodia and other governments hoped that this would end up turning into direct support from the Japanese government of FLOSS activities in the target countries, but this never materialized.

Meanwhile, IDRC (International Development Research Center, Canada) created in 2003 the PAN Localization project, in theory aimed at creating localized software in eight Asian countries. The program supported FLOSS in several countries, but in the specific case of Cambodia, it supported the creation of computer products for Windows platform, none of them in Khmer, and none of them released for free at the time.

5.3.4. Planning Khmer computing

The original version of the KhmerOS project was designed on the second half of 2003 by Javier Solá, a Spanish computer scientist who was traveling through Cambodia. The goal of the project, from the very beginning, was to ensure that the lack of software in local language would not be a barrier to the usage and development of ICT in Cambodia. The project included the localization (translation and adaptation) of software, its documentation, dissemination, and training of prospective users. As it was not possible to do this localization work with proprietary software – which only the rightsholders can modify and adapt – Free/Libre/Open Source Software (FLOSS) was chosen for the project.

The project first planned to promote the use of FLOSS Khmer language tools in the proprietary Microsoft Windows environment, and then gradually push for change to the use of the FLOSS platform Linux using Khmer language as a base.
After attempting – and failing - to create a consortium of private IT companies that would be interested on having this work done, KhmerOS became a project inside Open Forum of Cambodia (OFC), an NGO. With some funds from the NGO, and a few donations from private individuals, the project got some equipment and was able to hire two computer scientists in February 2004, starting the first real part of the work.

5.3.5. Finding support: a community, donors & government

From the very beginning KhmerOS attempted to create a local community of users, through its website: www.khmeros.info. Started in 2004, the website now has more than 6,300 registered users, and serves over 160,000 pages per month, with active forums, serving as a reference for the local FLOSS and local language computing community.

From the very beginning, finding funding was a priority. A grant from the Small Grants Program (several donors headed by UNDP-APDIP) was awarded to the project to write a FLOSS Localization Toolkit in which the experiences of the project were shared. This grant would be later followed by small grants from the Internet Society and UNESCO, giving the project enough funds to hire four more members for the localization team.

In 2005, InWEnt Capacity Building International, a German development aid agency, started supporting the training activities of KhmerOS with advice and funding, through its training project it@foss. Also in 2005 Javier Solá created in Spain - together with localization specialist Alberto Escudero - the WordForge Foundation, an organization aimed at supporting what it defined as “Digitally Endangered Languages” www.wordforgefoundation.org). The new foundation would become instrumental in securing – since 2006 – funding from the Spanish Agency for International Cooperation for Development (AECID). This would become the largest donor of the project, helping ensure that its results had the desired impact.

In 2004 the Ministry of Education, Youth and Sport, with the support and help from UNESCO, was preparing its ICT policy. The participation of the Open Forum of Cambodia led to ensuring that both the use of Khmer language in ICT and the use of Free and Open Source would be recommended in the policy document that was finally approved in January 2005. In 2005, Open Forum was approached by National ICT Development Authority (NiDA, the inter-ministerial body for ICT Development). A collaboration was started that would make the KhmerOS initiative a joint project of its parent NGO and NiDA. Several things came out of this collaboration:

- A Master Plan for deployment of FLOSS in Cambodia was developed. The plan was published (as a draft) and has been used as a reference for actions to be taken; parts of it have been adopted in the 2009-2013 Master Plan for ICT in Education
- KhmerOS members were involved in the improvement of the National ICT Policy, ensuring that Khmer language, Free and Open Source and Open Standards were part of it.
- A standard keyboard – based on the previous work by KhmerOS, and the first keyboard for use in Khmer – was defined as the NiDA Standard Unicode Keyboard, and publicized.

In 2005, with the support of the it@foss program of InWEnt, KhmerOS prepared and printed a Khmer user guide for OpenOffice (the FLOSS office document software application, localized to Khmer by the project), and started an ambitious training plan of government officials, computer teachers, NGO workers, and students in Phnom Penh and several provinces. They were all trained on the use of Khmer language FLOSS applications. In 2006 KhmerOS organized a National Typing contest, aimed at encouraging the learning of Unicode typing. Local contests were held in province capitals, and then a Nation-wide final was held in Phnom Penh, with the best three typists from each province. Spanish development aid from AECID started funding the project in 2006, covering all the needs that had not been covered until then for lack of sufficient funds. From 2007 onwards KhmerOS continued giving support on the use of Khmer software to government bodies, responding to their demand. The Ministry of Information, the Ministry of Culture and Religion, the Ministry of Interior, the Ministry of Agriculture, the Ministry of Rural Development, the Ministry of Woman's affairs, the Royal Palace, the Senate and the National Assembly have had their staff trained on the use of Khmer language FLOSS applications by the KhmerOS team. Open Institute has also trained computer scientists from the Ministry of agriculture, staff from NGOs, universities, and computer distributors on the administration of the FLOSS operating system Linux for use on file servers and Internet servers. As a consequence of these actions, the Royal School of Administration has started to teach the use of FLOSS to its students.

With cooperation and support from the Open Institute, the Cooperation Committee for Cambodia (CCC), the largest association of Cambodian NGOs, has started its own training program for NGO workers on the use of FLOSS in NGO's. The program became self-sustained in 2010.

5.3.6. Cooperation with other countries

KhmerOS has produced, from the very beginning, documents for replication of the project in other countries, as well as supported other collectives who wanted to do work on Khmer language. The localization effort also drew on efforts from other countries, such as translate.org.za (a FLOSS localization project for 15 South African languages funded by the Shuttleworth Foundation). KhmerOS has developed a localization project for the Tetum

language in East Timor, supported a similar effort in Uganda, and given direct technical support and/or advice for localization in Laos, Vietnam, Myanmar, Bhutan, Nepal, Bangladesh, Tanzania and - to a lesser degree – to other countries in Asia, Africa and Latin America.

KhmerOS members participate directly in some Open Source projects, such as OpenOffice, where they have provided code for the inclusion of the languages of several countries in the program, as well as manuals for how to do the technical localization of the OpenOffice code.

In 2005 KhmerOS started the WordForge project, dedicated to produce localization tools that used all the know-how that we had both accumulated on localization. The goal of WordForge was to facilitate the localization process in other countries that wanted to follow a similar process. The WordForge Foundation was created in Spain to find funding for this project. KhmerOS continues to working on this tool now with developers from Bangladesh and Spain, and this has led to the most advanced tool available for FLOSS localization, using Open Standards to produce high-quality language translations with volunteer and/or minimally-trained translators.

In 2007 KhmerOS/Open Schools Program was a finalist of the Stockholm Challenge/GKP Award in the Economic Development category. This prestigious international award has been given by the city of Stockholm since 1999 for the world's best initiatives using ICT for development.

5.3.7. Conclusion

The KhmerOS project shows that localization of Free and Open Source software produces sufficient added value - in countries in which there is not other software in the local language – for change to the use of FLOSS to take place. Together with the Open Schools Program it also shows the need to work on ICT policy as the vehicle that will lead the change. While national ICT policy must be affected, the impact of Education ICT policy is the real path to change, as it affects what users will get used to and use in the future.

The Open Schools Program has shown how a strong government/NGO partnership, with support from engaged donors and development partners, can produce effective policy and implementation of ICT in Education in a two year period, ensuring a clear path for a future in which support for use of ICT will be fully integrated in the Ministry, its overall five-year plans, and in its Annual Operational Plan and budget. Meanwhile, the Master Plan provides a guide for its development partners on the path that the Ministry wants to follow and for which it needs support.

Support from government is important to be able to penetrate society, and collaboration with international organizations is a good channel to reach the correct bodies of government. Localization of software without supporting materials (books and training materials) does not penetrate society, as resistance to change is strong. It is possible to start a project small, with few resources, but is also important later to be able to enlist sufficient financial resources for the development of training materials and for training.

Localization of FLOSS is a technical process that can be done in any country, if the necessary resources are available. Any materials, terminology or any other type of resources can be obtained. Staff who is proficient in their own language and have sufficient knowledge of English can act as translators, with checking the local language being the main skill required.

While still at an early stage, BanglaOS – the replication effort in Bangladesh – demonstrates that the model can be reproduced in other countries in a much shorter period of time of what was required in Cambodia, by working in parallel on localization and on policy, collaborating from the beginning with the national Education system.

5.4. IT@SCHOOLS: COMPUTERIZING STATE SCHOOLS IN KERALA, INDIA

Case summary	
Geography	India (Kerala)
IPR Issues	Open source licensed software adaptation & deployment: the project involved the use of existing open source software and the adaptation and development of software released under open source licenses.
Stakeholder incentives	Primary incentive for stakeholders (Kerala state government, school teachers & teacher trainers) has been the adaptation of software to local needs and cost-effectiveness, for which open source software was the most appropriate
Sustainability	The major costs are operational – teacher training – and are part of the general education budget of the government.
Impact	As the leading example of state-wide computer training in schools, this project has had a high impact.
Transferability	As with most open source software school deployment efforts, this drew on previous knowledge and cases. One well documented related case was the adaptation, development and widespread deployment of Open source software across schools in the Spanish region of Extremadura. However, despite the technical similarities in the software applications, wide region-to-region differences remain in the political, economic and organisational structure, which is the main part of any ICT-in-schools effort.
Public policy implications	Supporting initiatives that use open source in education, with the involvement of school teachers & teacher trainers, can have significant impact on software access & use, local software skills development, and local pride & sense of ownership and achievement, at relatively low cost

The Kerala IT@school programme provides computer education and computer enabled education through FLOSS tools to 1.6 million students annually in 2,738 high schools across 14 districts in the state, covering the last four years of schooling (grades 8 to 12 in the Indian system).

Initially, a training program was to be based on proprietary software. Following public protests from teachers and others, the state government reconsidered the use of proprietary software, in particular based on the argument that using it for training would make the education system dependent on monopoly vendors. Basic changes were made to the program to support the goal of universal access, including: providing large-scale in-house teacher capacity building programs; combining learning computer skills with computer-enabled learning in other subjects; and giving ownership to teachers to experiment with open source educational software in the classroom. Thus, instead of relying on vendors and investment in infrastructure, Kerala has chosen to invest in teacher capacity building on FLOSS, thus leading to the creation of an open source software eco-system.

5.4.1. Greater focus on computer aided learning

The teacher training program was designed to make the school teachers acquire basic computer literacy and open source educational software, but also to use computer aided learning to teach their own subjects. Teachers were also trained to install software and maintain hardware, making teachers comfortable with using computers. The rich availability of FLOSS educational tools and their provision to the schools under the program, coupled with teacher training, has enabled computer aided learning.

Well-qualified external experts trained an initial set of master trainers, who then trained their teacher colleagues. This removed the need for the external experts to be continuously required to train the entire teacher community. This saved costs, but perhaps more importantly, provided teachers with a sense of ownership and control of the program.

The teacher training systems, which are fully responsible for the pre-service and in-service training of teachers, were also responsible for training teachers computer skills. Since the training faculty is within the state education system, it ensured computer proficiency was developed and maintained as part of the on-going teacher training process. (It is important to note that the government education system in India has one of the largest, if not the largest, pool of teacher trainers in the world - there are more than 80,000 teacher trainers at cluster, block and district levels, whose primary responsibility is teacher training, both in-service and pre-service. Most of these teacher trainers or educators have a degree in education and have teaching experience in schools).

Significantly, this shift has also changed the nature of the ICT in schools program from being a centrally designed and implemented, with external resource persons, to owned by the schools, and supported by the school system. As an evaluation of this program noted, this was consciously in line with the philosophy of free software: it ensures the freedom to the school and the teacher to develop the curriculum and pedagogical methods the way they want to, which ensures their complete ownership and enthusiasm in the program.

5.4.2. Use of FLOSS educational software

Using regular in-house teacher trainers meant that open source knowledge was closely adapted to the needs of the teachers. Open source educational software is best used by teachers who understand the subject matter being taught, not just the software. More than technological expertise, what is required is that teachers can explore the software and determine for themselves how best to adapt it to their curriculum.

This process of contextualized ICT education by teacher support system allows for teachers to integrate computers into their own regular subjects, converting the computer from being a 'subject of learning' to 'process or method of learning' which took the program to much superior level of quality. This is seen from the continuous enrichment of the learning processes through the relevant use of additional tools. The 'school wiki' program has trained teachers in publishing digital content on the web to allow each school to have its own wiki page for sharing its work and ideas. This is also keeping in line with the collaborative philosophy echoed by FLOSS.

5.4.3. Systemic capacities for teacher education

The ‘Education Technology’ (ET) wing' in the District Institute for Education and Training (DIET) has the responsibility of understanding the role and possibilities for the use of technology in the school system. Making computer training an in-house integrated activity of

the school support system serves as an opportunity to increase the specialisation of ET faculty within the DIET.

Making the ET faculty responsible for the training for ICT in schools, including providing them with in-depth understanding covering the role of ICTs in learning and in society will strengthen the role of ICTs in the education system, making computer learning an integral part of the learning processes in schools. This also adds to the stature of the teacher educators as trainers in this 'new' arena of educational resources, methods and processes.

There is little justification in having only ICT training outsourced (to, e.g. private vendors) when all other kinds of educational training is done in-house, within the public teacher training system. If ICT education is seen to be a critical learning area, there is all the more reason to integrate it with the core of the education system, and use the existing capacities for in-service teacher education, instead of outsourcing the activity. This also implies that computer learning programs need to prioritize the needs of teacher educators and build their capacities for them to be able to work with teachers and schools, and this teacher preparation needs to precede the implementation of ICT in schools.

5.4.4. Free and customizable software

The Kerala project has made a significant effort in aligning the introduction of ICT to the learning contexts of the schools. Firstly, the department realized that office automation software (while important to learn) was not really the primary application for schools and that education required a larger set of software tools and applications that teachers and students could use and adapt for their own learning. The constructivist learning approach emphasized by the National Curriculum Framework 2005 specifies that learning happens not when the learner is merely the object of predetermined learning material, but requires the active engagement of the learner with the medium itself. These two imperatives – a large set of software tools, and the necessity of the learner to actively engage with these tools, led to the realization that proprietary software platforms would not suffice. Such platforms would not allow the learner to rise above the level of an 'end user', with no involvement in understanding the 'tools' and possibly 'co-constructing' them. Moreover, the pay per license model of proprietary software would make computer education enormously expensive, and unjustifiable in the context of a country like India.

Kerala's education department thus wanted to begin with a customized software distribution that would be relevant to, and appropriate for, its schools. While most computers come preloaded with Microsoft Windows and a few other applications such as Microsoft Office, with an English language interface, the department realized that this would not meet its goal of building in a large set of contextual educational applications, with local language interfaces. The choice of Free and Open Source Software (FLOSS) was thus logical. A FLOSS based approach could allow the department to take an existing software set and customize it in two ways – make the software interface completely available in the language spoken in the state (Malayalam), and to also bundle in hundreds of educational applications all available on a free and open source model along with the basic operating system.

The completely 'in-house' developed process and software design has also meant savings of millions of rupees that would have gone to vendors in the usual 'PPP' models, and these savings have supported the investments in further building in-house capacities for shaping new educational processes and curriculum using digital technologies, the role and scope of

which in any education system will only keep increasing. According to a recent study[47], the Government of Kerala saved around 500 million rupees ($11 million) as a result of opting for FLOSS. Even more importantly, FLOSS by reducing the costs of acquiring a computer helps in the faster and cheaper dispersion of computers outside the schools, in the homes of the students. Students and their parents are able to take the software used in schools and use it at their homes without having to either pirate proprietary software or pay huge license fees. This model also helps prevent complete dependence on technology vendors as well as resist marketing pressures.

5.4.5. Educational and local language software

Similar to the situation in Cambodia, South Africa, Bangladesh and elsewhere, the adaptability of open source software to local languages is directly related to its increased adoption. Schools in Kerala find the application interface in the local language, Malayalam, compatible with their medium of instruction. A local language software distribution has been made possible due to the conscious choice of free and open source software which has enabled the government to customize applications in the local language, and equally importantly to make available large number of educational software applications available to all schools at practically no cost. Students are therefore not limited to learning only office automation applications – which most typically associate with 'learning computers'; they engage with computers on a variety of areas from mathematics to science to environmental sciences.

The software distribution was customized from the publicly available Debian GNU/Linux operating system. The popular Edubuntu distribution which is specifically aimed at schools is also derived from the same Debian distribution and has hundreds of educational applications inbuilt. The issue of license fees / free sharing is not restricted to the operating system or office applications, but extends to educational resources. Educational software and content offered by large education technology companies is usually on a per-user license fee basis, which would make scaling and replication expensive. The Kerala SIET has created more than a thousand films on different subjects and provided them to schools for the 'digital libraries'. These can be freely copied and shared as required at marginal costs equaling just the cost of media.

5.4.6. Factors favoring a FLOSS eco-system in Kerala

It is worth exploring specific factors in Kerala that contributed to the success of the open source model in the state. First was the involvement of teachers' unions, who were consulted in the design and roll-out of the program. This helped get a greater support and buy-in of the teachers in implementing the program and in getting support and participation of the teachers for FLOSS. Teachers found installing and using FLOSS simple and did not want the program to use proprietary software. Second, the fact that most schools in Kerala have reasonable teacher-pupil ratios meant that schools could spare teachers for participating in the computer training programs and have one teacher in each school designated as a “computer teacher”. Third, the teacher training institutions of Kerala are also well staffed and could take on the responsibility of learning and teaching FLOSS on computers.

---

47 Rahul De, 2009. *Economic Impact of Free and Open Source Software – A Study in India.* Indian Institute of Management, Bangalore. Available at: http://www.iimb.ernet.in/~rahulde/RD_FOSSRep2009.pdf

While the above mentioned set of factors may be within the influence of any public sector education system, there are other factors which are perhaps unique within India to the state of Kerala. These include very high levels of literacy, greater urbanization, higher availability of transport, communication facilities and electricity. Kerala's 'Akshaya' program of the IT Mission in Kerala, which created computer infrastructure in villages across the state, in the form of tele-centres, and provided basic computer literacy to one member of each household, would also have helped in providing local capacity building and hardware / software support. The political-ideological inclinations of the left-of-centre government in the state could also be a factor that favored the spread of FLOSS in the state, although open source policies have received support across the political spectrum in India[48].

5.4.7. Curriculum – a critical factor of the FLOSS eco-system

Curriculum design played a key role in the success of the Kerala program. The implications for pedagogy and learning arising from a casual approach to ICT and ICT-based curriculum include both making computer learning largely unconnected to the larger curricular design of the education system and not leveraging the best FLOSS possibilities for learning. Vendor-driven or product-driven ICT policies are typical in many deployments of software – and related references to software in curriculum.

In Kerala, a vendor-driven approach was consciously excluded. Instead, the curricular content for the program was created through workshops with regular teachers and educationists were clearly in charge of the process. The program supports the development of curricular material by teachers in each school, with school “wikis” providing a grassroots, interactive and collaborative content creation process at the local level.

5.4.8. Exploring new possibilities for learning through FLOSS educational software:

Education through computers in schools has enormous possibilities. Providing access to a wide variety of information sources (reliance on the single text book is an acknowledged limitation of learning possibilities in schools), connecting students to peers and other learning community members (which would transcend space and time), creating new digital artifacts and publishing / sharing the same, are some new possibilities that can significantly impact learning processes. (At the same time, there are new skills that may be required to be learnt, for instance, learning to discriminate and identify authentic from spurious sources of information, which would be a component of critical pedagogy, defensive access to the internet to protect against 'virtual predators' etc.) However for any of these possibilities, it is essential that the entire system of learning be grounded and integrated in the mainstream education system and its design and implementation driven by the members of the system itself - comprising of teachers, teacher educators, students and educationists. The collaborative scope of FLOSS allows this.

Over time, the outcomes of the efforts of the vendors and technology experts would become the default curriculum which can have negative implications for learning. Use of FLOSS in schools removes the dependence on external vendors, thus giving complete ownership to the school and its teachers.

5.4.9. “Public Software”: using terminology to facilitate non-technical discourse

[48] E.g. the 2009 manifesto of the right-of-centre BJP party: http://public-software.in/BJP-IT-vision and that of the left-of-centre CPI(M) party: http://public-software.in/CPI%28M%29-manifesto

As a follow-on to the IT@Schools project in Kerala, and through the discussions and interactions in the EU-funded FLOSSInclude project, the notion of "public software" emerged as a term, and strategy, for the use of and access to software for the public good. This has been the subject of workshops and discussions highlighted on the Public Software portal hosted by the NGO IT for Change, and has helped provide political and policy-maker support for public access to software. As in the case of public education or public health, public institutions are, following this argument, responsible for ensuring access to public software as well as support public participation in its creation and sharing.

"Software developed for public service has a unique context and objectives deriving from those of public service; with its imperative of providing public goods and ensuring equity and social justice. It is well known that private and commercial actions have very different context, motives and considerations than public actions. For instance, the largest possible reach and diffusion as well as transparency of actions are basic to public service, which are not necessarily values espoused by private and commercial players. Thus public software would cater to the requirements of universal access, transparency and participation. Public Software being publicly owned, allows for its free sharing as well as modification by all. Public Software is thus Free Software. In addition, public software is also a public good. While Free Software requires the freedoms of the individual user to use, study, share and modify the source code, in addition to this, public software emphasizes its 'public good' nature and vests on government the responsibility of ensuring that basic software required for negotiating the digital world is freely available to all."[49]

5.5. OPEN SOURCE OBSERVATORY AND REPOSITORY (OSOR): FACILITATING KNOWLEDGE SHARING AND COMMUNITY BUILDING IN EUROPE

Case summary	
Geography	Europe-wide
IPR Issues	IPR issues were not faced directly by the project, which acted to facilitate the release of software developed by public administrations under open source licenses. However, the project also acts as a competence centre, publishing studies directly addressing IPR issues that stakeholders might face – choosing licences, copyright issues, interaction between open source and patents, etc.
Stakeholder incentives	The project was funded by the European Commission, with the aim of increasing software sharing among public authorities and across the public sector in general. Incentives for other key stakeholders were: gaining recognition for local initiatives and access to peers in other administrations (for administrations contributing and participating in project activities, including sharing their software on the OSOR portal); single-point access to a public sector software sharing community (for open source developer community, civil society and industry stakeholders)
Sustainability	Operational sustainability for keeping the OSOR.eu portal going is relatively low in terms of physical infrastructure costs. Much of the content and all hosted software is provided by the (mostly public sector) software rightsholders. During the initially funded period, a knowledge

[49] "What Is Public Software", available online at: http://public-software.in/Public-software

	base of answers to IPR and other issues was built up and is maintained by the community of participants, who also contribute to news updates, the most visible part of the portal.
Impact	OSOR.eu has acted as a major catalyst in coordinating open source initiatives in Europe and is one reason for Europe's global preponderance in this area (see the table of regional distribution of initiatives in the previous chapter). It is now the world's biggest single source of news updates and case studies on public sector open source software, and its software portal is similar to several others worldwide.
Transferability	Several efforts worldwide have implemented the model of a hosted community of public sector open source software, though not necessarily as broad in scope as OSOR.eu which had a big focus on facilitating cooperation across different countries and building a knowledge-base, in addition to cataloguing software and catalyzing its release under open source licenses. Softwarepublico.gov.br is an example of a parallel initiative in Brazil.
Public policy implications	Building or supporting initiatives that aggregate and disseminate information about open source software use can increase software development and increase sharing of software and reducing costs in the public sector

Following on the successful Open Source Observatory initiative in 2003-2005, which published regular news reports and case studies on open source software use, deployment, and development in public administration in Europe, the Open Source Observatory and Repository (OSOR) was initiated in late 2006. It was designed as a pan-European collaborative environment to federate public sector Open Source developments. It was designed to include a Repository – a site where software packages and information about software can be hosted, providing a "home" for software that has been released under open source licences and therefore may be legitimately acquired from sources unconnected to the rightsholders).

The point of OSOR was to encourage the re-use of publicly-financed software through the use of Free/Libre/Open Source Software (FLOSS) distribution and deployment, by becoming:

- A pan-European information platform on FLOSS: providing news, guidance, links, contacts;
- A platform for uploading and downloading software produced by and for public administrations;
- A platform/"forge"[50] for cross-border collaboration providing technical, organizational, and legal support.

Volunteer collaboration in producing free or open source software is nothing new, as the movement was initiated in the eighties. Technical environments allowing doing so were also developed early (SourceForge.net is the most famous, and has many derivative versions).

[50] A "forge" is an online web-based platform where software under an open source licence can be stored, downloaded, maintained and modified, while keeping track of individual contributions and modifications through sophisticated version control systems. The term "forge" comes from Sourceforge.net the first widely-used such platform.

In the past few years, the EU recognized that this form of collaboration has extended to the production of software by organizations – such as companies, and to a limited extent, public administrations – and was no longer limited to individual volunteers. Indeed, organizations may account for at least one third of open source software available today, possibly much higher for some projects such as OpenOffice.org, Linux or Apache[51].

Meanwhile, the public sector in Europe accounts for some 20% of the ICT market[52]. At the same time, while about 29% of software investment in the EU is on in-house software development (and a further 53% is on custom developed software)[53], it is apparent that much software spending in the public sector is duplicative in nature. Providing mechanisms for pooling and sharing such software would reduce costs, increase efficiencies and increase collaborative innovation in the public sector. In this context it is remarkable that over 10% of local government authorities in the EU stated that they own software that could be released under an FLOSS licence[54].

From this, the European Commission saw a clear potential for a public service, such as the OSOR, to enable the sharing and shared development of software by and for the public sector in Europe. The OSOR aimed to not be only a platform for software development – a forge – but to bring together an accompanying effort to provide service, support and community-building synergies addressing the specific needs of the European Public sector. Due to national, linguistic, cultural and legal barriers, only a small amount of transnational collaboration has been undertaken in the field of software used by the public sector. The OSOR as a public service would aim to change this.

The Open Source Observatory and Repository project could be perceived from different points of view. From the European Commission's point of view, OSOR started as fully funded project (IDABC - DG Digit 2006-2009) in contrast to the many research projects that are initiated by various groups, e.g. through grant funding or other sources. OSOR faced high expectations from the EU authorities and all stakeholders, as it was seen as the most ambitious and significant support from the EC to develop an innovative knowledge society in the specific domain of public sector software.

From the economic point of view, the OSOR aimed to reduce the duplication of effort that comes from different public administrations developing software for the same tasks, in effect re-inventing the wheel. This was seen as likely to save taxpayers' money in the long run, making the OSOR a service that is not only in the public interest, but also provides a substantial – if indirect – return on investment.

From a strategic or policy point of view, the OSOR project could be seen as a potential driver for changing software development and distribution policies in both EU, national and local public administration and for facilitating the implementation of “free/libre/open source” ecosystems around software used and produced by the public sector.

---

51 DG Enterprise, “Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU”, January 2007

52 According to an estimate by Dr Tech Kari Tilli, Director (telecommunications and electronics industries) of Tekes published in March 2006 by the European Commission – see http://europa.eu.int/information_society/research/vienna_process/vienna_documents/documents/k_tilli.pdf

53 See data from the FISTERA network cited in table 24, page 124 of R. A. Ghosh, “Study on the economic impact of open source software on innovation and the competitiveness of the ICT sector in the EU” – www.flossimpact.eu

54 DG INFSO, “Effect on the development of the information society of European public bodies making their own software available as open source”, Published on http://www.publicsectoross.info/ (July 2007).

More concretely, from the end users' (or beneficiaries') point of view, the OSOR was to provide a *service* dedicated to the common needs of specific public sector communities that was not previously provided at a single location. It was therefore important to identify concretely the target stakeholders and to focus on some specific groups or stakeholders that could act as pilots or models for others:

- Existing open source repositories in EU Member States that might be interested in forming a network (a repository function) and,
- FLOSS projects associated with public authorities interested in using the OSOR service as exchange or development platform (a collaboration function).

The OSOR acted as a Competence Centre for the multiple emerging initiatives in Member States (and inside the European institutions), by extending the work of the pre-existing Open Source Observatory (OSO): providing regular news, events and newsletters, cases studies and reports providing legal and strategic advice. This served to actively assist public bodies about the use and collaborative development of FLOSS. Reports published covered topics such as patents and public sector use of FLOSS; building links to between public administrations and FLOSS developer communities; and the influential and widely cited "Guideline on public procurement of Open Source Software"[55]. The OSOR also resulted in the creation of the European Union Public Licence, an Open Source licence with legally valid translations in all official EU languages and determined to be in full compliance with EU law. This *imprimatur* made a big difference to public administrations – who are conservative, but nevertheless may lack sufficient legal advice – in terms of reducing concerns about open source software licensing.

A key innovation of the OSOR repository was the federated search – the ability to search for software hosted on the OSOR directly, but also those hosted on repositories supported by individual regions, Member States, or independent initiatives. Using the European Commission's expertise and services for translation, this service allows users to search in any EU language; keywords are translated into the languages of affiliated repositories; software descriptions in search results are than re-translated and collected for the user. As repositories are as much about community building as physically hosting software, the ability to interact with and collate from other repositories was important – the OSOR did not intend to replace other, more local initiatives, but to facilitate them and facilitate interactions across them.

To this end, the OSOR includes a collaborative workspace to develop and share experience and software where necessary: a Wiki-based collective memory, a "forge" to support collaborative software development, mechanisms to put public administration in contact with one another through OSOR user groups, forums for collaborative discussion, the facilitation of specialized improvements and adaptations, software localization or certification, support for the development of ecosystems around public sector software (such as technical support, or other services).

The OSOR now has 80 published case studies and several hundreds of news items, several times a week; over 200 software projects are hosted directly on the OSOR, with a further 2500 searchable through its federated repository system. The European Commission initially planned the project as a 4-year trial, expecting that it would be self-sustaining. However, its widespread impact and stakeholder demand has led to a decision to continue supporting it,

---

[55] Available online at http://www.osor.eu/idabc-studies/OSS-procurement-guideline%20-final.pdf

integrating it as a key part of the European Commission's ISA (Interoperability Solutions for European Public Administrations) program.

5.6. SOFTWAREPUBLICO: BRAZILIAN GOVERNMENT SOFTWARE PORTAL

Case summary	
Geography	Brazil
IPR Issues	IPR issues were not faced directly by the project, which acted to facilitate the release of software developed by public administrations under open source licenses.
Stakeholder incentives	The project was funded by the Ministry of Planning, Budget and Management, with the aim of increasing software sharing among public authorities and across the public sector in general. Incentives for other key stakeholders were: gaining recognition for local initiatives and access to peers in other administrations (for administrations contributing and participating in project activities, including sharing their software on the OSOR portal); single-point access to a public sector software sharing community (for open source developer community, civil society and industry stakeholders)
Sustainability	Operational sustainability for keeping the Softwarepublico.gov.br portal going is relatively low in terms of physical infrastructure costs. Much of the content and all hosted software is provided by the (mostly public sector) software rightsholders. The Brazilian Ministry of Planning, Budget and Management pays the costs of running the portal.
Impact	Since its creation in 2007, the portal has grown to a community of 130 000 registered users, with software in solutions in different areas and multiple awards for e-government and innovation.
Transferability	Several efforts worldwide have implemented the model of a hosted community of public sector open source software.
Public policy implications	Building or supporting initiatives that aggregate and disseminate information about open source software use can increase software development and increase sharing of software and reducing costs in the public sector

The Brazilian Public Software Portal is a space for providing IT solutions to the public sector

According to the Brazilian Ministry of Planning, Budget and Management[56], "*More than 5,500 municipalities spread across a continental region, limited resources, cultural diversity and a need for integrated operations are some of the factors that have promoted the use of open software in Brazilian public administration. In this scenario, and based on very successful experiences in the use and licensing of open software, the Brazilian government created the Portal do Software Público Brasileiro [Brazilian Public Software Portal] to systematize the combination of resources and create a single source of solutions in open software for public administration, especially at the municipal level*".

[56] This case description is drawn from correspondence between the author and officials of the Brazilian Ministry of Planning, Budget and Management; as well as the Ministry's publication on the Public Software Portal.

The Portal was created as a place to share software between government authorities, civil society and the non-profit sector; software on the portal is all distributed under a reciprocal FLOSS licence, the GPL, ensuring that it can be accessed, studied, modified and redistributed and that all modifications remain accessible under the same terms.

In addition to providing access to specific software solutions, and allowing the upload of new solutions and improvements in already existing software, the portal also receives contributions from users and organizations in fields such as quality, professional training, financial support, management and international communication. Similar to the OSOR.eu in Europe, the Brazilian Public Software Portal goes beyond software to provide a community, with resources such as articles, interviews, links and a guide to service providers that is continuously growing.

The software is expected to be a finished solution that is ready to install and use, fully documented, like any commercial off-the-shelf software. A set of basic user services is provided by the Portal, including various interactive online forums for discussion, feedback and support, a version control tool for modifying the software and tracking modifications, and system documentation.

The community is supported by a technical team; management and control tools are defined to establish the frequency of the release of new versions and provide quality control parameters for on-going software development. The Portal also provides for a uniform process of availability for any entity or individual that participates in the model, guaranteeing the release of the software, the continuity of the project and the functioning of the ecosystem.

According to the Ministry, the software solutions, which can be adapted for use in any country or organization, provide various benefits including:

- Rationalization of resources – the sharing of solutions reduces the replication of development efforts;
- Immediate availability – the solutions, which are documented, can be downloaded free of charge, through simple registration on the portal;
- Sharing of administrative experiences among municipalities, and analogously, of technological knowledge and of rules of negotiation among the members of the user communities created around the solutions;
- Creation of business opportunities for local IT companies, which are dedicated to customization and improved solutions;
- Sharing of improvements – developments and corrections can be made available on the portal;
- Choice of supplier – the user can opt for the service provider and for the contracting model considered most suitable.

The Portal is seen by the Ministry as a success and a key part of Brazil's public sector software infrastructure. It was created in April 2007, starting with just one software application hosted, and now offers dozens of solutions in various areas (education, processing of geographic information, computing, administration and healthcare). It now has more than 130 000 registered users.

6. CONCLUSIONS AND RECOMMENDATIONS FOR WIPO'S ROLE

This Study set out to investigate the use to which copyright law can be put to facilitate the application of software development practices to economic, social and cultural development,

in developing countries and LDCs. To this end, the Study has examined the general treatment of software in copyright law at the international level and national and regional regimes. The primary conclusion from this initial exercise is that software has been treated as primarily an industrial activity, rather than, say, a form of essential knowledge or information to which access should be given some consideration. Unlike, say, for educational materials, public information or even (with relation to patents) pharmaceuticals, legislative and regulatory practices for software do not in general provide for exceptions or limitations to rights provided.

General practices within copyright law, such as limitations and exceptions, do apply to software too. They have had a limited impact on software development practices. The one software-specific exception applied in copyright law is the "interoperability exception", incorporated in the EU Software Directive and related to Article 40 of TRIPS, which allows for appropriate treatment of IPR licensing practices that may be considered anti-competitive. Again, in practical terms, this has a limited impact on the software market in terms of increasing access for the purposes of economic and social development.

One important reason for the lack of legislative or regulatory initiatives towards improving access to software through copyright exceptions has been the development and success over the past two decades of an alternative software development model that does not rely primarily on the economic exploitation of exclusive rights over software. Free software, also called libre software or open source software[57], is a phenomenon that has grown from small beginnings in the academic community in the 1980s to powering the majority of devices and services people use to connect to Internet today. This study has shown how open source software, while providing an alternative software development model and supporting a range of business models for economic exploitation of software and related services, works within current copyright regimes. Indeed, key features of popular open source software licenses rely on copyright law for their functioning, and open source licenses have been enforced through copyright law in the courts.

Open source software, while functioning within the traditional copyright regime, greatly increases access to and the ability to participate in software development. As explained in this study, it does this through an economic model and a licensing framework emphasizing the sharing of information based on voluntary copyright licensing mechanisms chosen by rights-holders, rather than through legislative actions around copyright law. Methods used by governments to facilitate software development for economic development are, therefore, typically been in terms of increasing demand, supply or broad access for open source software. Demand-increasing measures, such as procurement policies that result in an increase in the public use of open source software, have the result of making open source software more economically sustainable for local businesses (and may increase the efficiency of public spending on software). Supply-increasing measures include policies to release software developed for or by the public sector under open source licenses; measures to fund the development of open source software, either directly or through fiscal (tax) treatment of contribution to open source software development. Broader access-increasing measures include training programs that encourage entrepreneurial activity around open source software; facilitation of information sharing among public sector, private sector and community developers; educational access programs such as the use of open source software in schools; ICT access programs, such as adaptations to local languages through the use of open source software models.

[57] Free/Libre/Open Source Software is generally referred to in this document by the acronym FLOSS, a term used in a number of studies and policy documents in Europe, Africa and Latin America.

Unusually, open source software is an access-increasing model that has until now been more successful in wealthier countries, with adoption driven largely by business demand. This is not surprising – with some exceptions, there is a high geographic correlation between open source software developers and access to computers and the Internet[58]. Partly as a result, a wealth of empirical evidence has been collected over the past few years on the economics of open source software, and this Study has examined this material – albeit with a particular focus on the development of local knowledge and skills, and local economic development. Open source software has been shown to have a strong impact on access to software and more importantly from the perspective of long term sustainable economic development, access to skills development and participation in the software creation process. A summary of policies and initiatives regarding open source software has been provided in this Study, based on data from dozens of countries around the world. A more comprehensive survey of official open source-related policies is cited in the bibliography, and distributed as an annex to this report.

A few especially interesting cases going beyond policy – not necessarily initiated by governments – have been examined in more detail. They show how open source models for copyright have allowed public initiatives to rapidly develop, access and deploy software systems with significant impact. Initiatives examined were sometimes originated by government, but often originated by civil society or industry and later supported by public organizations – underscoring the flexibility of open source licensing, which allows users and developers to bypass the transaction costs and times typical of traditional copyright exploitation models. The cases selected emphasize how the open source copyright model can be used not only to increase *access* to software in developing countries (as passive consumers) but how this model is suited for the creation of software innovations in developing countries. Examples such as the use of Sahana (originated in Sri Lanka) in New York or Ushahidi (originated in Kenya) in New Zealand show how the open source model allows developing countries to actively participate and contribute value in the global software development community, providing a flow of knowledge that may be surprising.

Policy strategies focus mainly on correcting current policies and practices that implicitly or explicitly favour proprietary software. Some of these policy strategies were recommended (and have since been followed in whole or in part) in the recommendations of the European Commission report “Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU”, January 2007.

Recommendations are listed below with a primary focus on WIPO initiatives, and a secondary focus on member states activities:

1. Avoid aggravating policy lag; increase awareness of open source as a source of innovation in software. WIPO, like many member states, has in the past not paid much positive attention to open source, although this has changed in the past few years. The world's software industry is now, with few exceptions, run on open source software. The global economy – the New York and London Stock Exchanges and NASDAQ – run on open source software (Linux)[59]. Mobile computing, the fastest

[58] Note that in terms of open source *usage*, developing countries do not lag; indeed, if mobile computing is included, open source software runs at the core of most smartphones and more advanced feature phones whether in the form of Linux, Android (essentially Google's version of Linux) or even Apple's iPhone, which is build like all modern Apple systems on the open source FreeBSD platform.

[59] See http://www.computerworlduk.com/news/networking/3244936/london-stock-exchange-smashes-world-record-trade-speed-with-linux/

growing way for ICT distribution in developing countries, is also dominated by open source software such as Linux and Google's Android[60]. It is important for policy makers at all levels to recognise that open source licensing is an innovation and licensing model that has been widely accepted by industry and provides a legitimate way for broadening ICT access.

2. WIPO should include open source licensing and IPR issues in technical training. Unlike many sectors of IPR, broadening access to software does not necessarily depend on exceptions and limitations; open source software relies on copyright law within the boundaries set by TRIPS. National PTOs and copyright offices often lack awareness of the IPR issues involved with open source software; as it is an important policy option, WIPO should ensure the provision of technical training to increase knowledge and awareness among member states. Several resources for this purpose have been created by member states themselves (especially within the EU's OSOR project)

3. WIPO should specifically address open source in discussions on standards and IPR, specifically Standards Policy and Patent Policy, where open source software may be penalised. Recent publications and policy statements in the EU (specifically, European Commission) are highly relevant for an appropriate approach

4. Encourage the study of fiscal policies, such as equitable tax treatment for open source creators: open source software contributions should be treated as charitable donations for tax purposes. Where this is already possible, spread awareness among firms, contributors and authorities. (Primarily an issue for member states, although perhaps also for WIPO – the ToR for this study specifically called for a discussion of fiscal issues)

5. Avoid penalising open source in innovation and R&D incentives, public R&D funding and public software procurement that is currently often anti-competitive and favours specific proprietary brands to a far greater extent than most other sectors of procurement (member states)

6. Avoid lifelong vendor lock-in in educational systems by teaching students skills, not specific applications; encourage participation in open source-like communities (member states)

7. GLOSSARY OF COMMON ACRONYMS

FLOSS: *An acronym unifying the terms free software, libre software and open source software, all of which refer to software that is distributed under the terms of the free software definition or open source definition, which are equivalent. Free/Libre/Open Source Software, and the acronym FLOSS, is used in a number of studies and policy documents in Europe, Africa and Latin America.*

GPL: *GNU General Public License, the most commonly used copyright licence for distributing FLOSS, used for GNU/Linux, Android, and several other FLOSS software systems.*

---

[60] Even Apple's iOS and Mac OS X run on an open source software operating system, with a proprietary graphical user interface.

ICT: *Information & Communication Technologies*

IPR: *Intellectual Property Rights, including Copyrights, Patents and Trademarks*

NGO: *Non-Governmental Organization, typically non-profit organizations*

SMEs: *Small or Medium Enterprises, businesses with a small or medium number of employees and revenue.*

[End of document]