

OPEN SOURCE SOFTWARES: AN OVERVIEW

Written by *Akanksha Sikri**

**5th Year BA LLB Student, Amity Law School, Delhi (GGSIPU)*

Introduction:

In the recent years open source software¹ has altered the contour of software industry. Open source and commercial software are the prominent models of software licensing. Open source software (OSS) licensing model, unlike the conventional proprietary software² development model of limited access, allows users of the software under certain terms and conditions, to redistribute, modify and add to the software program. OSS also emphasizes the users of the software on unrestricted accessibility and availability of source code. OSS has been developed through a collaborative effort in which larger groups of people interact and contribute the elements of the final software. So, Open source Software (OSS) can be defined as “software whose source code is openly published, is often developed by voluntary efforts and is usually available at no charge under a license defined by the Open Source Initiative which prevents it from being redistributed under a more restrictive license.”³ This means that software and technologies developed under an OSS license are available in source code format, allowing the modification by anyone who finds a need to do so, as well as the redistribution of OSS without financial or legal ramifications (provided this is done within the scope of the applicable OSS license). An example of an OSS license is the “General Public License (GPL)” authored by the Free Software Foundation⁴ that today governs the use of a huge proportion of all OSS released today. In brief, it states that software licensed under the GPL will remain free under any circumstances, and that it cannot be appropriated by anyone in a closed/proprietary manner. It also mandates that anyone is free to distribute the software, and may modify it as required, provided that such modifications are given back to the community so that others may benefit

¹ Software does not have any universally accepted definition yet and may cover in its ambit computer programmes, computer databases and may include items produced by the operation of a computer programme such as documents, drawings and other works stored or transmitted electronically or even printed out on paper.

² Refer Appendix-I.

³ Open source Software Policy, UK Govt.

⁴ (<http://www.gnu.org/licenses/gpl.html>).

from the modifications. Conventional software development is governed by IP laws which were predominantly practiced over the past few decades. The open source movement necessitates scrutiny; more than just being a new-fangled approach, it catalyzes debate regarding both the mode of software production and its protection. It is being used to propel arguments to revisit IP jurisprudence. After all, the intention of the intellectual property-software system is to catalyze innovation and ultimately serve the society.

Open-Source Software and Intellectual Property Rights:

Cyber laws Inventions, discoveries and technologies widen scientific horizons but also pose new challenges for the legal world. Information technology brought about by computers, internet and cyberspace has also posed new problems in jurisprudence. These problems have arisen in all areas of law. The law (statutory or otherwise) providing answers to these problems or dealing with information technology are sometimes loosely referred to as computer laws or e-information technology laws or simply cyber laws. Intellectual property rights (IPR) are an important aspect of cyber laws. In Intellectual property rights (IPR) what is worth copying is prima facie worth protecting. These rights refer to the property that is a creation of the mind: inventions, literary and artistic works, symbols, names, images and designs used in commerce. It is broadly divided into two categories: - Copyright, which includes literary and artistic works such as novels, poems and plays, films, musical works, drawings, paintings, photographs, sculptures and architectural designs. It lies in description of some. - Industrial property, which includes inventions (patents), trademarks, industrial designs and geographic indications of source. In India, IPRs are protected under different Acts namely, The Copyright Act, 1957; The Designs Act, 2000; The Geographical Indications of Goods (Registration and Protection) Act, 1999; The Patents Act, 1970; The Protection of Plant Varieties and Farmers Rights Act, 2001; The Semiconductor Integrated Circuits Layout Design Act, 2000 and The Trade Marks Act, 1999. Apart from the aforesaid Acts, IPRs are dealt with in two more areas, namely, the Trade Secret Act and the Indian Contract Act.

Trade Secret Protection:

Open source deals with sharing of software's designs and components among the users. The concept of trade secret as a legal tool is to ensure protection for software. However, considering the success rate of open source software, protecting ideas as trade secret pose a problem for

developers, thus leading to an inference that sharing ideas in software domain delivers higher benefits to its developers. Some of the disadvantages of protecting source code as trade secrets are insufficient protection, data security and user's privacy risks, fewer incentives to developers. On the other hand, some of the benefits of OSS development model are competitive improvisation, skills enhancement, user-driven innovation, etc. Hence, OSS model proves to be more socially efficient whereas the trade secret model offers the developers a choice between bearing the costs of inefficient software development and monitoring a dominant firm to prevent possible abuses. Therefore, the scope for trade secrets form of IPR and OSS to co-exist is a rare possibility.⁵

Copyright Protection:

According to copyright law, the original author/creator owns the exclusive right to reproduce a work by default, subject to certain conditions. The owner of a copyright can license or sell the right to copy his work to others under certain terms and conditions. Hence violating certain terms and conditions lead to infringing on the copyright. 'Copyleft' is a term widely used in open source licensing. Copyright laws are used by an author to prohibit others from reproducing, distributing or adapting copies of his work. Contrary to the term 'Copyright', 'Copyleft' allows an author to reproduce, adapt or distribute copies of his work but the resulting copies are bound to certain license agreements. Thus copyright laws enable the creators of OSS to make code available through an open source license usually the General Public License. In *SCO v. IBM* case¹⁶, SCO claimed that IBM infringed their copyright and trade secrets, by illegally incorporating SCO's proprietary UNIX code into open source Linux operating system and subsequently demanded that Linux users needed a license from them, for parts of the Linux code. The SCO Group sued a number of companies for donating UNIX code to Linux. The court ultimately gave a verdict that 326 lines of code in Linux kernel were potentially infringing. From the case mentioned above, it is evident that enforcing an open source license is using copyrights laws in full force. Therefore an open source license is obviously using copyright laws to protect the way the author's work is used. Moral Rights are very much applicable to open source mainly because the development of such a model is highly dependent

⁵ SK Verma, IP Protection of Software and Software Contracts in India: A Legal Quagmire, *Journal of Intellectual Property Rights*, Vol.17, July 2012, pp 284-295.

⁶ *Caldera Sys Inc v Int'l Bus Mach's Corp* (D.Utah 2003) (No. 03-CV-0294).

on the individual contribution of users/developers of the community as a whole. In certain countries, where moral rights of the authors are legally recognized, violation of open source licensing terms would constitute a violation of the author's moral rights. Though the GPL covers a broad scope of protection, the modifier's action can still be restricted by the author based on Section 14⁷. By resorting to such a provision, the author retains a right to prevent any impairments of his work.

The Copyright Act as this has bearing on open source software. Copyright: source code and object code computers do not understand our language. They only understand machine language or machine code i.e. instructions which consist of a series of 0s and 1s. In the earlier days, a computer program used to be written in machine code or by punching a punch card. The punched slot or un-punched slot indicated requisite information to the computer. This process was slow and tedious. Such a program, although intelligible to the computer, was virtually unintelligible to anyone except an equally skilled programmer. From earlier days, the computer scientists also devised an alternative language for writing programs, known as assembler language. These assembler languages had advantages over writing a program in machine code but they still required many instructions to be written in order to achieve the simplest tasks. A number of high-level languages “such as Basic, Fortran, Cobol, Pascal etc.” have been devised in order to simplify the work of a programmer. The use of these high-level languages enables a programmer to write a program in terms that nearly resemble ordinary English unlike those used in the lower-level languages. They also permit complex operations for the computer to be directed by a relatively compact command. The programs as written by a programmer are known as the source code. When an assembler or a compiler converts it into machine code, it is known as the object code. Generally, this conversion is one-way from source code to the object code. However, it is possible to reverse it but de-compilation and disassembling is time-consuming and expensive. GAIM is a popular program for it loads different instant messengers (MSN or Yahoo) together. Source code of GAIM is open; it is known to everyone. It is written in C++. If one reads it one can understand a few words mentioned therein and what it is trying to say. It is a kind of description of something and it amounts to literary work within the meaning of the Copyright Act and is so protected. A source code of a computer program if it is open is a literary work within the Copyright Act. However,

⁷ The Copyright Act, 1957.

often it is not disclosed and is kept as a trade secret. But is an object code also protected? The question, whether source code and object code both are protected by copyright or not, has troubled the courts and has been differently answered. The Australian High Court in 1986 held that the source code is a literary work and is protected as a copyright. But no such protection was given to the object code. One of the Judges in the majority held: "I have not found anything that has persuaded me that (the object code) a sequence of electrical impulses in a silicon chip not capable itself of communicating anything directly to a human recipient, and designed only to operate a computer, is itself a literary work, or is the translation of a literary work within the Copyright Act. The aforesaid question did not arise in India. Earlier the provisions in the Copyright Act in our country were similar; it was possible that courts might have rendered similar judgments. Amendments in the Copyright Act; The Berne Convention for protection of literary and artistic works in 1986 provided that computer software (object code and source code) and compilation of data be protected under the Copyright Acts. Agreement on Trade Related Aspect of Intellectual Property Rights (TRIPS) is part of the charter establishing World Trade Organization (WTO). Every member of WTO (and our country is a member) has to accept it. It has proceeded from the Berne Convention and Article 10 of TRIPS requires members to amend the laws accordingly. Since then we have amended the Copyright Act by two amending Acts namely Act 38 of 1994 and Act 49 of 1999. These amending Acts amended Section 2(o) of the Copyright Act to change the definition of the words literary work, it now includes computer program as well as computer database. The result is that not only the computer programs (subject code as well as object code) but computer database is also protected as a copyright. In India infringement of a copyright is a penal offence and civil remedies (injunction, damages etc.) are also available (TRIPS Articles 41 to 50, 61). By the two amending Acts consequential amendments were also made in other sections to make enforcement more realistic. For licensing in open-source software, one never purchases software but merely takes a license to use it. There are different kinds of licenses. Software where source code is not open and is secret is known as close-source software i.e Proprietary software and is a close-source software, where in general the user has only limited right to use a product, on a specific computer (sometimes with a specific power or processor) sometimes with a limited number of signed or concurrent users. Freeware is usually used when a piece of software is given at no cost. Generally the programs are released only as executables, with their source code not available. For example, one can download the Adobe Acrobat Reader as

a freeware, but the software is still proprietary. Shareware is usually distributed free of charge for a limited period of time or for a limited use, mainly to give the user the opportunity to test it before buying it, try before you buy is their motto. Public domain software is a software for which copyrights do not exist. Although this notion is invalid in Europe (but can be understood under US law), often used for software; anyone can use it for any purpose, without any restriction. However, the availability of the source code is not guaranteed. Copyrights are used to protect computer software but everyone is not using copyrights to have rights in software. Some are using copyrights so that no one may have any rights in software; there is a new word for it i.e. copyleft. Software where source code and object code both are freely available is called 'open-source software' (OSS). In copylefted open-source software, source code and object code are freely available to be used, modified and improved without any charges. Not all open-source software is copylefted; it could be non-copylefted. This depends on the terms of license of the software. The term open-source software is often used in the sense of open-source software that is copylefted. In order to copyleft a software, the owner first states that it is copyrighted and then adds distribution terms that gives everyone the right to use, modify and redistribute (the source code and the object code in the original or modified program) only if the distribution terms are unchanged, modified version (source code and object code) is freely available and could be further modified and distributed only on the same terms. In other words, there is freedom to modify the software and anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it. Copylefting guarantees that every user has freedom. There is some debate whether open-source software (copylefted) be called free software. One section [Free Software Foundation (FSF)] says there is freedom to modify software and it should be called free software. However, this causes confusion as the word free has the meaning that it is without any cost. It could be mistakenly understood for the software that is of no cost or freeware. The other section [Open Source Initiative (OSI)] prefers to call it open-source software. The ideals of both are the same; it would be better if they could sort out this difference. Using of the words open-source software rather than free software just to avoid confusion. This does not diminish the contribution of either of them. General public license (GPL) contains a condition that copyleft a software. GNU is a recursive acronym and stands for Unix. So, open source would rather use the legal weapon of copyright as an invitation to join in the fun, rather than as a weapon against others. It's still the same old mantra: Make Love, Not War, except on a slightly more

abstract level. Imagine an intellectual property law that actually took other people's rights into account, too. Imagine IP laws that encouraged openness and sharing. Laws that say sure, you can still have your secrets, whether they be technological or religious, but that doesn't mandate legal protection for such secrecy.

Patent Protection:

Software patents are criticized extensively by the open source community. OSS model is designed to disseminate software license as widely as possible. Open source proponents seek to revisit patent jurisprudence in the context of software programs altogether, citing it as an 'undeserved reward'⁸. Free and open source licensing has several implications such as targeting wide audience and sprawling rights that multiply with the every adaptation of the software. Even, several technology companies and its developers discourage software patents owing to the fact that software inventions are too abstract and closer to mathematical algorithms than concrete industrial machines. Free and OSS community envisages a model which involves the practice of writing and releasing software codes freely. Such a model, offers a license to its licensee, the right to modify and redistribute the software. When software is distributed by someone in such a fashion, depending on the terms of the license the recipients will gain an identical sub-license from the distributor or a direct license from original licensor. All free and OSS licenses permit free adaptation of the material. Such a license also allows for free distribution of these adapted materials. Some licenses such as GNUGPL, has made it mandatory that the adaptation of the software it covers must be distributed under the same license which is termed as 'copyleft'. From the aforesaid, it is evident that free and OSS licensing model provides a cheaper route for use and distribution of potential licensees who do not want to pay for a license. Release of source code disqualifies the opportunities to obtain patents on processes embodied in the code Thus, the patent form of protection is not a popular choice among the practitioners of OSS community owing to the aforesaid difficulties as well the delay in implementing patent sharing.

Trademarks:

⁸ Rowe K, Why pay for what's free?: Minimizing the patent threat to free and open source software, John Marshall Review of Intellectual Property Law, 7(2) (2008)595.

Open source developers exhibited sophistication in the use of trademark law instead resorting to certification marks so as to indicate that a particular software comply with the requirements of open source scheme for example 'OSI Certified' mark affixed by the Open Source Initiative for software complying with their Open Source definition. The term 'Copyleft' by itself has acquired a distinctive trade use. Therefore, the use of certification marks ensures greater flexibility and avoids several hassles like policing of the mark and ensuring proper arbitration and imparting equal responsibility on all the developers⁹.

So, earlier computer programming was done on one-to-one basis and internet was still established as a medium of connectivity. However, it is not the case anymore. Software is rarely individually tailored, but mass produced as a team work collaborating with global teams. Open source development model has become a practical alternative to commercial/proprietary software. Therefore, OSS model plays a superior role in software industry. Further, open source offers a bundle of incentives by reducing IP benefits. However, compared to patents, OSS model expedite discovery through automatic disclosure. It is to be noted that such a model is not viable and cannot operate in every other field where patent benefits dominate.

Licensing and Transfer of OSS:

For more than three decades, the transfer of technology to countries of the South has been a recurring theme in various multilateral discussions focused on economic development, the stimulation of further innovation, foreign direct development, and, more recently since the late 1980s, on the spread of the global intellectual property regimes. This debate, especially its intellectual property component, took on a higher profile during the lengthy GATT discussions of the Uruguay Round that culminated in the TRIPS Agreement of 1995. Probably the most traditional argument for IPRs protection in developing countries is that technology owners are less willing to transfer proprietary knowledge to countries with weak protection because of the risk of piracy. A corollary or consequence of this logic is that strengthened, indeed globally harmonized, intellectual property protection would result in significantly increased levels of technology transfer to countries of the South. And although some countries of the South expressed certain reservations about this logic, indeed numbers feared that increased

⁹ Vikrant Narayan Vasudeva, Open Source Software Paradigm and Intellectual Property Rights, Vol. 17, Journal of Intellectual Property Rights, November 2012, pp 511-520

intellectual property protection would, instead, limit access to technology and hinder innovation in their countries and regions they signed on to what has been labelled in some circles as one of the TRIPS trade-offs in the mid-1990s. Why? While there is ongoing academic and economic debate about the precise role of expanded levels of technology in the overall developmental process, there is little debate that such countries possess sub-optimal levels of technology and that this weakness is one of the key barriers to their economic growth and international competitiveness. In exchange for agreeing to significantly higher levels of intellectual property protection and enhanced enforcement measures, countries of the South successfully argued that TRIPS must contain language mandating the transfer of technology. Articles 7 and 8 of TRIPS, found in that agreements basic principles under Article 66 (2), mandate the transfer of technology to least developed countries. Subsequently in the Doha Ministerial Declaration of 2001, the World Trade Organization agreed to create a working group that was tasked to examine the relationship between trade and technology transfer and of possible recommendations on steps that might be taken within the mandate of the WTO to increase the flows of technology to developing countries. The creation of this working group was an agenda demand of such countries because, five years after the signing of the TRIPS Agreement, there had been very little evidence that industrialized countries were acting on their TRIPS commitments to promote and facilitate the transfer of technology. In the Africa Group proposals of 4 October 2001 for an alternative text to the Draft Doha Declaration, these nations asked that developed country Members shall put into immediate effect meaningful incentives for the purpose of promoting and encouraging technology transfer. Today and more than two years after Doha, there have been no significant improvements in the levels of technology transfer from North to South. Does it remain a matter of only tertiary concern to industrialized countries? Technology transfer issues unfold with particular clarity in the case of computer software. Moreover, the differences between the two types of software, proprietary and FLOSS are stark. As an initial matter, the legal acquisition of proprietary software by users in the South (or the North) is generally not conducted as a sale; rather, such software is usually licensed, increasingly on an annual basis and requires, therefore, the payment of annual license fees. But this legal transfer is not a technology transfer. The definitions of technology and technological transfer contained in the UNCTAD draft by International Code on the Technology Transfer excludes goods that are sold or hired [or leased] from the ambit of technology. Thus it is the knowledge that goes into the creation and provision

of the product that constitutes technology, not the finished product or service. It is precisely as a finished consumer product that its manufacturers, wholesalers and retailers conceive proprietary software; the licensing of such software is therefore outside the ambit of generally accepted notions of North-to-South technology transfer. What is particular about the licensing of proprietary software is that the user does not get access to the all-important source code; rather, the source code is completely inaccessible to that user and protected by the twin legal regimes of copyright and trade secret law (and perhaps also by patent law). And the licensors intention is that it be kept that way as a non-accessible secret that is not shareable with others. The licensing terms make this explicit. It is analogous to the sale of a tractor engine by an equipment manufacturer to a Southern farmer, except that the farmer is not allowed, by law, to look inside that engine and understand how it works, change the spark plugs him or herself, improve its performance, modify it to perform tasks that the vendor does not wish it to perform, or, for example, figure out how a cheaper and more efficient version could be manufactured in his or her own country. Even though the licensing of such software is not a technology transfer, even if it was, one of the key transfers of technology issues is whether the technology which is transferred is capable of local adaptation. The source code of proprietary software is not adaptable because, in the first place, intellectual property protections make it unavailable. By comparison, FLOSS requires that the source be made available, in other words, capable of adaptation and assimilation to local conditions for all who want or need it (or at least under the terms of the GPL and other similar licenses.) Miguel de Icaza, a Mexican-born open source software developer who now is president of Boston-based Ximian, explain that the beauty of free software is that part of the freedoms you receive is the freedom to learn from other people's techniques, strategies, and focus on problem solving.¹⁰ So people have a chance to join the effort, and be part of the team of people that are producing knowledge, culture and, as a result, wealth. It is this unrestricted access to the source code which not only creates the potential for IT sector to grow in developing and least developed countries, but also allows users and other software developers to create their own software tailored to their own needs and their own national and regional languages. FLOSS, by permitting access to its source code, allows users or members of users group to faulty programmes and builds self-reliance in permitting them to do their own repairs and servicing. By comparison, proprietary software does not permit the

¹⁰ Alan Story, Intellectual Property and Computer Software, Intellectual Property Rights and Sustainable Development, Issue Paper No. 10, May 2004.

user to make his or her repairs and to attempt to do so may void the guarantee. To return to the tractor analogy, the FLOSS allows the farmer, who may be quite remote from a repair shop, to diagnose faults himself or herself or to call upon nearby farmers to assist in the repair process and allows others to learn from the repairs. This self-help approach, not permitted or feasible with closed source proprietary software, can dramatically cut ongoing computer usage costs, a factor of significance importance for poorer nations. In other words, FLOSS permits, indeed facilitates and encourages, technology transfer. Lacking such adaptability, proprietary software results in mere product diffusion.

Technological transfer viability of OSS:

The inflexibility and technical biases of proprietary software further lock countries of the South into a pattern of dependency and economic stagnation; FLOSS systems promote technological self-reliance and independence. Again, here is how ‘Miguel de caza’ puts the case for the role of free software in countries of the South believing that Free Software will help countries with developing economies (like Mexico) to get a competitive advantage that they have lacked for so long. Most of these countries missed the industrial revolution, and for one reasons or another, they depend on external technology to keep up with the times. Free Software helps in the fraction on depending on external technologies. For example, countries with developing economies can now avoid depending on proprietary software: they can keep the money they spent on proprietary software to themselves, and use it to either develop themselves, or they can use that money to produce free software that will solve their problems (and hopefully other countries problems). The case of Mexico is the one I am most familiar with: Mexico does produce very little technology, depends a lot on foreign technology and pretty much our main exports are raw materials. Raw materials are extremely cheap (and in some cases it took nature a few million years to produce). For example, a barrel of petrol costs about \$25 these days, and a copy of Microsoft Office and Microsoft Windows 2000 costs around \$700. This means that for each copy of Office+Windows 2000 the country is paying with 24 barrels of petrol. In general, we must become software producers (and also technology and innovation producers), and not just consumers. Becoming free software users is a good first step; the next step is to become software producers. Another open source pioneer, Ivan Moura Campos, prime developer of Brazil;s Popular PC project, believes countries such as Brazil will not overcome the so-called digital divide by relying solely on imported technologies, such as copyrighted

proprietary software. Another example of the problems endemic to closed source proprietary software, especially in operating systems, is that a company such as Microsoft is permitted to bundle a wide number of other computing products into its Windows (and now Windows XP) operating system. Such practices not only capture the global market in operating systems but also control many of the ancillary activities related to day-to-day computing as well. As one commentator has explained: The nub of the case against the company is this: why should it be allowed to bundle products like media player into its operating system for free instead of being required to distribute them as stand-alone extras? Who knows how much innovation, especially from smaller firms, has been stifled at birth because of the impossibility of competing with what Microsoft is bundling for free? Once again, if software developers in technologically advanced countries such as the US and Europe cannot compete with Microsoft in the application program market, how will software companies in the South? When one proprietary operating system such as Windows becomes the operating standard in countries of the South, no forward internal economic linkages are created and a minimum of wider IT economic development is generated. Any technology that is transferred is primarily an internal transfer to affiliates under its control and the retention of technology and skills within the network of the trans-national corporations may hold back deeper learning processes and spillovers into the local economy, especially where the local affiliate is not developing R & D capabilities. The principal consequence of this monopolized global proprietary standard in software operating systems is that when companies such as Microsoft expand to new locations in the South, that expansion primarily means the establishment of a local sales office and the hiring of sales and clerical staff and software installers, but few of the much more skilled software writers and programmers. The new Cybercity project in Mauritius is one example of this phenomenon.

Suggestions¹¹:

The Government of India (GOI) must adopt a comprehensive and supportive open source policy. It builds on their earlier efforts to adopt open standards for procurement. As we've seen in other regions, the adoption of such policies often brings out concerns from some quarters

¹¹ Extracts taken from Atul Chitnis of Exocore Consulting (P) Limited, and is partially based on a forthcoming whitepaper being prepared by Exocore for submission to the Government of India.

who want to spread 'fear and doubt' about the policy. So, what are the facts about the policy, and how does it fit into India's broader economic development strategy? From 'purchaser' to 'innovator' many governments are increasing their engagement on open source to help them promote a culture of innovation that they need in order to serve their citizens today and in the years to come. While government procurement regimes often lag behind those in the commercial sector in terms of adaptability and efficiency, there is a growing awareness among not only the IT experts but also the leadership of the public sector that the old way of acquiring software has to change and that lock-in is no longer acceptable. The use of technology, including open source software, is moving out of the sphere of simply 'acquiring a product' to 'investing in innovation'. As a result, the emphasis on open source by the GOI is a reflection of this change of focus. IT is less about acquiring intellectual property via a license, and more about widely distributing the tools and adding value on top of it. This paradigm shift has enabled decision-makers to go from thinking of small 'procurement' windows to viewing open source from a broader vantage point that highlights its broad-based benefits to an economy, jobs, and innovation, and in the government itself. In India, the policy coincides with another broad initiative, Digital India. By bringing together various functions and efforts, the program seeks to prepare India for a knowledge future. It is centered on three key areas: Digital Infrastructure as a Utility to Every Citizen, Governance & Services on Demand and Digital Empowerment of Citizens. Those objectives are consistent with one of the other exciting trends I'm seeing: governments using open source software, as a key component of 'digital agenda' initiatives that include open standards and open data policies, to enhance civic engagement. Whether through sponsoring 'app challenges' or 'hackathons' to generate excitement around new ways of using government services and information, to modernizing online web-based services, and governments actually 'open sourcing' the software, there is strong evidence that open source is indeed driving transparency and better engagement with citizens. One example is the work of the US White House to connect citizens (and citizen developers) to government (and government data). The US "digital agenda" is carrying out the President's goal of using technology to make a real difference in individuals' daily lives. Notably, in carrying out its effort, the White House is committed to "using and contributing back to open source software as a way of making it easier for the government to share data, improve tools and services, and return value to taxpayers." In many ways, the GOI policy is consistent with these trends we see around the world. With this background, what's the new policy all about? The thrust of the

policy requires that the various ministries of the GOI "shall endeavor to adopt Open Source Software [OSS] in all e-Governance systems implemented by various Government organizations, as a preferred option in comparison to Closed Source Software (CSS)." To effectuate the policy, "All Government Organizations, while implementing e-Governance applications and systems must include a specific requirement in Request for Proposal (RFP) for all suppliers to consider OSS along with CSS while responding. Suppliers shall provide justification for exclusion of OSS in their response, as the case may be. Government Organizations shall ensure compliance with this requirement and decide by comparing both OSS and CSS options with respect to capability, strategic control, scalability, security, life-time costs and support requirements." Reportedly, the draft of the policy was sent to 82 government departments, and inputs were received from 65 departments. Most departments were reported to be very supportive of the policy, with none opposed the policy. More than 40 governments world-wide, by my conservative count, have policies that create a positive environment for open source use. These policies are important to level the playing field, not merely highlighting the benefits of open source to governments (saying it's ok to use it) but also providing meaningful answers to commonly asked questions by government IT professionals. The French government, for example, published a guideline in late 2012 urging the country's public administrations to not only make a thorough and systematic review of free alternatives when building and revising IT infrastructure and applications, but also to use the savings realized by the use of open source to develop expertise and engage upstream communities. The GOI approach is also consistent with the direction of the US Government. Just last year, the White House (via the Office of Management and Budget and the Federal CIO) issued a Digital Services Playbook—described in some quarters as "something of a marvel for an official government policy: it's elegantly designed, has clear navigation, and is responsive to any device you choose to view it upon." At its core, the Playbook is about more agile use of reusable software and processes that focus on the customer. Central to that approach is its emphasis on open source. The final 'play' in the Playbook captures the notion of 'Default to Open'. Play 8 encourages agencies to 'Choose a Modern Technology Stack' which focuses on use of open source, cloud-based, and commodity solutions across the technology stack, "as these solutions have seen widespread adoption and support by the most successful private-sector consumer and enterprise software technology companies." It clearly states, "Consider open source software solutions at all layers of the stack." Open source will continue

to be the 'go to' approach for governments around the world, and the steps that the GOI have taken are consistent with that trend. The policy reflects a potent driver toward open source software utilization: the fundamental shift in IT architecture, away from coupled hardware, software, and data to more modularity, reuse, and a central focus on interoperability—all of which is enhanced by tighter government IT budgets and the goal of avoiding vendor lock-in.

The counter arguments being that the GOI policy has received criticism, directly from some companies as well as from organizations representing some segments of the IT industry. Having observed attacks on open source policies for much of the last decade, these are different in style and tone. Remarkably, there is little dispute about the policy itself. Rather, the central thrust of the attack is on the 'how to' of implementation. Instead, as one of the organizations stated in its letter to the GOI, the "policy on this issue should emphasize open standards, interoperability and other key factors that create a level playing field for vendors of all types." Creating level playing field is precisely what the GOI policy is striving to achieve. And there is no doubt that use of open standards and ensuring interoperability are important. But they are not sufficient. The challenge is changing a culture and approach that provides the GOI with options. As it was indicated in an interview that "the objective of the policy was to ensure both the options [of closed and open source] are compared and the best possible solution adopted. The objective of the policy is not to narrow down the opportunities for closed source software companies, but to ensure that both CSS and OSS options are properly evaluated.¹²" Moreover, claims that the GOI policy 'mandates' OSS are also misplaced. Secretary Sharma went on to say: "It is clarified that the policy does not make it mandatory for all future applications and services to be designed using the open source software (OSS). The compliance part of the policy clearly states that the solution suppliers should consider OSS along with closed-source software (CSS) while proposing solutions. They can always propose CSS solutions, provided they can justify it over OSS. While the government organizations shall ensure compliance with this requirement and decide by comparing both OSS and CSS options with respect to capability, strategic control, scalability, security, life-time costs and support requirements." The GOI is to be commended by taking this initiative. It is essential that the GOI and the open source community, working with all vendors who support OSS, come together to raise

¹² As Ram Sewak Sharma, the Secretary of the Department of Electronics and Information Technology (Deity) [indicated in a media review](#),

awareness and ensure meaningful implementation of the policy. The Government of India, through the Ministry of Communications and Information Technology (Department of Information Technology), has requested inputs and recommendations for the the use of Free/Libre OpenSource software (FLOSS) and technologies such as Linux by government agencies, as well as in other spheres of operation in India.

“Linux” is a term used to refer to the following:

1. As the “kernel” (or core) of an operating system, around which an entire operating system is built. This is known as “The Linux kernel” as developed originally by Linus Torvalds and the development community.
2. As an Operating System (or OS distribution) that is based on the Linux kernel, incorporating many OSS components to provide a complete computing platform. Though the correct terminology would be “Linux+Gnu+Apache+....”, the most common usage in public mouth (and the best understood one) is “The Linux OS” or just plain “Linux”. (The actual definition is subject to innumerable debates, and is beyond the scope of this document).
3. As term representative of the Open Source movement in general. Linux is the most visible “poster child” of the movement, and has gained instant name recognition, with enviable “branding”.

“Open Source” is a term used to refer to the following:

1. The class of license used to license the software
2. The development methodology used in the development of software.

In this document, “Linux”, “Open Source” and “Free/Libre Open Source Software (FLOSS)” are sometimes used interchangeably. While this may not be correct technically, it does provide a certain amount of convenience to the authors.

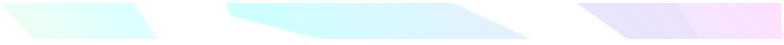
Issue 1: Government Stand

The government is considering the adoption of Open Source for its IT operations, as well as for the use in IT operations it influences, such as education. However, a question arises whether the government should mandate the use of Open Source software and technologies, locking out other sources of software. As expected, this move would be completely contrary to what is expected of a Government, which can (and should) offer recommended guidelines, and


naturally follow them itself, but not create a monopoly situation that does not differ from existing proprietary software monopolies. The Government could indicate its preference of Open Source software in its procurement process, but not at the cost of functionality. Instead, it should mandate a software/technologies evaluation, selection and procurement process that clearly includes Open Source software and technologies, comparing it on functional and financial basis to other offerings. The most important part here is “functional” – it would be illogical to reject a solution for which no functional equivalent Open Source offering is available. However, this is a favorite hunting ground for proprietary software vendors. It is easy to lock out an Open Source solution by specifying a functionality that is proprietary. An example of this would be “ability to read Microsoft Word documents without loss of formatting or content”. This is a trap, since only Microsoft Word actually knows the structure of the documents it creates, and this structure is a closely guarded trade secret. Instead, a requirement specification should read as “should be able to store documents in an open and well-documented format that makes it easy for other applications to access information contained in the document, without interaction with the original developer of the software that created the original document”.

Issue 2: Adoption of the products or the process?

Some of the tangible benefits of Linux:

- 
- No cost
 - Increased stability
 - Enhanced security
 - Better functionality
 - Adherence to published standards
 - Interoperability.

Certain conditions which needs to there are as follows:

- 
- Source available for all to view (increased security, better stability)
 - Source available for all to modify (better functionality)
 - Worked on by a large, diffuse team (Standard adherence, Interoperability).
 - Source and binaries free for all to distribute (No cost)

In other words, Linux offers the listed benefits precisely because it follows a Free/Libre and Open Source Software (FLOSS) development model. Without that model, Linux is just another also-ran in a long list of also-ran operating systems. It's a great mistake to separate out the benefits of Linux (listed) from the causes of those benefits (the FLOSS development model) and tout Linux as the solution keeps evolving to meet the needs of different environments and entities; sequestering the development model from the software will ultimately result in another operating environment that mirrors the (to take an example) Microsoft set of products and their inherent weaknesses. While Linux does represent the most visible of such products, it is the public development model (that includes the creation as well as review of such code) that is actually at the heart of matter. It would be insufficient for the Government to consider and adopt specific FLOSS products without using the development process as criterion.

Importance:

Already a huge number of products have started appearing that “run under Linux”, a measurable proportion of them are just another manifestation of the closed source model of development. This does not provide the government any benefits of the FLOSS model.

Recommendation:

While selecting products based on FLOSS, the Government should specify that “Open source products indicate products or technologies that follow the Open source methodology of development, and not just operate on an Open source platform.”

Issue 3: Support Services

A favourite “weapon” against Open source software is the concept of “support by the vendor”, and an implied “someone you can sue”. It has been implied that Open source software does not enjoy the kind of support provided by proprietary software vendors. This impression is completely wrong, and its propagation should not be encouraged. Almost all software support for proprietary software is a charged process. In fact, proprietary software vendors create commercial “training” and “certification” facilities for their products that have now become an entire industry of its own. However rarely (if ever) do these “support services” exceed the quality and availability of support for Open source software, as evidenced by the incredible growth and adoption of FLOSS in India.

Recommendation:

The Government can change the face value of “support services” in a very simple and effective manner – provide resources (and possibly funds and recommendations) that will ensure that support services for open source software in India. This can be done by encouraging the growth of technical and vocational training services in Universities and even schools, thereby providing greater scope for employment to students at all levels, without the financial penalties normally associated with the “training” in support for proprietary software.

Issue 4: Adoption and Policy

The Government will face an incredibly difficult task (made difficult by many commercial interests at stake for proprietary software vendors) formulating a fair policy that includes the use of open source software. Luckily, much of the hard work has already been done by governments of countries such as Germany, France, the United Kingdom, etc.

Since the functioning of the Government of UK is very similar to that of the Government of India, adaptation of these policies and procedures should considerably ease the process of formulation.

Issue 5: Resources

The single biggest advantage open source enjoys is availability of resources. Whether in the form of modifiable and usable source code for every OSS product, or support material – there is no dearth of such resources. Also, over the past few years, India has witnessed a tremendous growth in awareness of (and technical competence in) open source products. Pioneered PCQuest (whose Linux Initiative has been running since 1996, and has been responsible for the distribution of more than a million Linux CDs in India), many IT-related publications have made available vast amounts of technical information and other material to their readers over the years. Open source Conferences, such as India’s premier annual Linux/OSS conference Linux Bangalore (<http://linux-bangalore.org/2002>), which was sponsored and endorsed by the Government of India, have grown from small regional get-togethers to huge national and international prime events, recognized for their professional execution and content, and hailed for their usefulness to people interested in open source technologies. Many premier training institutions across India now offer Linux and related OSS courses. But most important of all, India’s technical education system is built entirely around the concepts of Unix, which Linux embodies today. In fact, most computer science subjects, as well as technical and vocational courses, use Linux and Unix as a base for their curriculum, resulting in vast numbers of

technically competent and qualified Linux & Unix savvy engineers emerging from universities every year. By recognizing open source Software as a desirable platform for Government and related IT operations, the government will prevent a situation where students graduating from our universities will have to spend a fortune in getting training in a proprietary technology before they become “employable”.

Hence, the vast Linux and open source community of India applauds the desire by the Government of India to adopt Linux and open source as a source for its technology fulfillments, and looks forward to working with Government to help it in this endeavor. However, the community does caution the Government that it is a rocky road ahead, given the number of policy and commercial issues that need to be considered, and it urges the Government to make use of the community’s support and participation in surmounting these obstacles. Because just like open source products and open source development processes, there is only one way to work on this project. In view of the above, there is a need to formulate a policy for the Government Organizations to adopt Open Source Software.

