The Open Source Revolution

The Competitive Edge for 21st Century Businesses and Local Economies

Aaron D. Harper
Jaye L. Sudar
Lobo Savvy Technologies, LLC.
Ahead Research, Inc.
Colorado Technical University, Colorado Springs
Ashford University

5 June 2010

A Whitepaper to Colorado's Workforce Investment Boards and Businesses

Abstract

Who could have predicted that a request for help would spawn a revolution. While the expense of commercial software has increased, an alternative has risen to the challenge of providing quality business solutions for less. Largely decried by opponents invested in the commerce of proprietary software, Open Source software has become a strong contender and good fit for any business need. Better still, it allows companies to choose when and where their dollars are spent, permitting social responsibility by supporting their local economy.

This revolution is the Open Source software movement, and its win-win philosophy may well be the competitive edge business needs to survive the turbulent economy of the 21st century.

Acknowledgement

This whitepaper is one of four prepared in preparation for the 2010 Rocky Mountain Workforce Development Association conference in cooperation with Lobo Savvy Technologies LLC, Colorado Technical University, and data supplied by Ahead Research Inc., the South Central Region Workforce Investment Board, and Spanish Peaks Library District.

Contact

Aaron D. Harper, aharper@lobosavvy.com
Jaye Sudar, jsudar@lobosavvy.com

Introduction

In 1991 a student in Helsinki Finland posted on a bulletin board "*I'm doing a (free) operating system (just a hobby, won't be big and professional like gnu) for 386(486) AT clones. This has been brewing since april, and is starting to get ready"* (Torvalds, 1991). He went on to rally help in his project, first locally at his school, then globally using the Internet. The project is still going strong with over 3700 active developers and many more reviewing and testing the code which is now both big and professional. It runs on over 60% of servers and 90% of the world's supercomputers. The project is the Linux operating system, and it has become a benchmark for Open Source software and the free software revolution.

The concept of Open Source is based upon the recognition of intellectual property, and the freedom to use, incorporate and improve upon it. This is neither a new, nor communist idea. In the early years of automobile development, a group of capital monopolists owned the rights to a gasoline engine patent originally filed by George B. Selden. By controlling this patent, they were able to monopolize the industry and force car manufacturers to adhere to their demands, or risk a lawsuit. In 1911, independent automaker Henry Ford won a challenge to the Selden patent.

The result was that the Selden patent became virtually worthless and a new association was formed. The new association instituted a cross-licensing agreement among all US auto manufacturers: although each company would develop technology and file patents, these patents were shared openly and without the exchange of money between all the manufacturers. Up to the point where the US entered World War II, 607 patents were used freely between competing American auto manufacturers without lawsuits or exchange of money. (Flink, 1977)

To extend the concept to software it must become clear that software is not the media used to distribute it, but rather the program, thus the intellectual property itself. This means that to truly allow full use of the software, it must be open to review, adoption, and improvement. This is a way for intellectual property to take on a life of its own and grow rather than molder away in a company safe. Everyone with the ability to program and an interest becomes a developer, or at least peer review. It became a truism when Eric S. Raymond, a vocal proponent of the Open Source movement and published writer stated in his landmark book *The Cathedral and the Bazaar* "...given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix will be obvious to someone." (Raymond, 1997) He called this Linus' Law in honor of Linus Torvalds, the original developer of the Linux operating system mentioned at the beginning of this paper.

At this point the world of software development and many other products which rely on intellectual property changed overnight. Instead of a choice between similar products for the highest price the market would bear, we now have a choice between the philosophies of Open Source and closed source. It his book *The Cathedral and the Bazaar,* Eric S. Raymond likens the development, management, and financial models to two venues. One where a product is carefully crafted by select people working in isolation, the cathedral, is contrasted to a large group of people working in an ad hoc collective to a common goal, the bazaar.

What is Open Source

To be considered Open Source, software must meet a set of ten very specific criteria (OSI, 2010):

1. **Free Redistribution.** The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.
2. **Source Code.** The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.
3. **Derived Works.** The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.
4. **Integrity of The Author's Source Code.** The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.
5. **No Discrimination Against Persons or Groups.** The license must not discriminate against any person or group of persons.
6. **No Discrimination Against Fields of Endeavor.** The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.
7. **Distribution of License.** The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.
8. **License Must Not Be Specific to a Product.** The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.
9. **License Must Not Restrict Other Software.** The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.
10. **License Must Be Technology-Neutral**. No provision of the license may be predicated on any individual technology or style of interface.

More specific criteria are part of the various software licenses used in Open Source. There are as of the time of this writing 66 recognized Open Source licenses coverings various products and developer groups. (OSI, 2010) The most common Open Source license, the GPL, is in Appendix C at the end of this whitepaper.

Advantages of Open Source Software

The developers mentioned previously are one of the greatest assets to Open Source. Their contributions to Open Source software are generally unpaid and thus the motivation to produce the quantity and quality of code for the project is done as a type of ethic rather than as a result of a standard reward system of money and position. To put it clearly, beyond the basic needs of food, water and shelter, a person with this ethical system will value duty and camaraderie more than more money or power. In this the Open Source developer eschews both the work centered capitalist thinking and the similar one in the communist mindset, as both derive from the protestant ethic and look remarkably similar from this detached neutral perspective. Both are alien to what is now called the Hacker Ethic. (Himanen, 2001)

What this means to Open Source software is that nothing is truly done for the money or power. Instead, accomplishments are an end unto themselves, and the methods and paths chosen reflect this with a profound effect on the product. The differences can be highlighted in five basic areas which we will examine in turn:

1. Reliability and Stability
2. Auditability and Security
3. Cost
4. Flexibility and Freedom
5. Support and Accountability

Reliability and Stability

Much has been said about the reliability and stability of Open Source software, both pro and con. First, let's define reliability. It is the "ability of a computer program to perform its intended functions and operations in a system's environment, without experiencing failure (system crash)." (Business Dictionary, 2010). Because of the number of eyes on the ball, the number of bugs are reduced, and the sheer number of developers make for very rapid fixes. "Severe defects tend to be fixed within hours of their being detected, a process which is undoubtedly assisted by the availability of the source code. Able developers who discover a bug will commonly also fix it and then report it to the maintainers as well as issuing an updated version of the software on their own authority" (GB direct, 2006).

The end result is that while malfunctioning software is fixed within the community within a short time, often hours for major bugs, the official roll out of the fix often takes as long as that of a closed source offering. This encourages IT support personnel in organizations which use Open Source software to be active in the developer forums to speed the development of fixes to the world and deployment of code patches to their organization and clients.

Another factor is stability. Where a tool has been created that fits the need well, it doesn't change without a good reason, and furthering marketing goals doesn't factor in at all in Open Source development. That does not mean Open Source software development is stagnant. Quite to the contrary, minor changes happen on a very frequent schedule. These changes usually happen behind the

scenes leaving the user interface intact with menu options and screen metrics right where an experienced user expects to find them.

This unique development schedule is a little confusing to some. In contrast to closed source software which is assumed to be production ready code (sometimes erroneously), Open Source software is released publicly from the moment of inception. This means that the task of determining suitability is the responsibility of the one selecting the software.

One hint the developers give us is the version number. While the numbers are fairly arbitrary, there is some level of standardization. Any version number below 1.0.0 (eg. 0.7.2) is not considered production ready by the developer. This does not mean such software isn't ready for use. Developers often have a very strong ethic which demands perfection prior to calling it version 1.0.0. For example, Inkscape is a vector drawing program similar to Xara, Adobe Illustrator, and Corel Draw. It is in production use by many organizations and is as usable as its commercial, closed source counterparts, even though the latest version at the time of writing is 0.4.7.

Auditability and Security

Auditability and security are the biggest advantage of Open Source software. Anyone can view the source code and are encouraged to do so. On the surface it would seem that software which is so transparent would be inherently insecure, and this is one of the arguments proposed by Open Source opponents. In practice the level of peer review tends to make Open Source software more secure, not less. This is one of the reasons the Linux operating system, MySQL database, and other Open Source projects have been selected by many governments and their intelligence organizations around the world. Many not only use Open Source software, but are generous contributors.

One notable example is the National Security Agency of the United States, NSA. This shadowy organization has arguably done more for Linux security than any other contributor simply because they chose to release the source code of some of their internal modifications. “The architecture has been subsequently mainstreamed into Linux and ported to several other systems, including the Solaris™ operating system, the FreeBSD® operating system, and the Darwin kernel, spawning a wide range of related work.” (NSA, 2006)

Before adoption into the main Linux tree the contributed code was carefully examined by every developer concerned over security and privacy. They were able to go through the code line by line because of Open Source's transparency. Review and derivative work is even encouraged. “There is still much work needed to develop a complete security solution. Nonetheless, we feel we have presented a good starting point to bring valuable security features to mainstream operating systems. We are looking forward to building upon this work with other developers and users. Participation with comments, constructive criticism, and/or improvements is welcome.” (NSA, 2006) Collaboration such as this between a commercial or government entity and the Open Source developer community has made the Linux operating system arguably the most secure operating system on the planet, which furthers NSA's goals of keeping America's secrets.

Cost

Cost is less of a factor than might be thought. While the “Hacker Ethic” tends to keep costs to a minimum by not feeding management egos with obscene paychecks, companies generally have no problem paying a fair price for a good solution. The Achilles heel of the Open Source model is that fewer potential users become aware of the offerings than their closed source counterparts. This is a marketing problem.

When it comes to marketing, it is clear and intuitive that commercial software companies have marketing dollars to spend, while Open Source solutions do not. This is because the Open Source solution is usually free, and the relatively modest amount of funding needed to support an Open Source software company is generated by services the company offers. In the Open Source business model, information is free, but expertise is not. These may have a person or group for public affairs, but generally no true sales department.

Further, other than some big companies like RedHat, IBM, Novell, and Oracle, the company which produces open source software will generally not have a global reach due to a limited budget. This means that support of Open Source Software falls to local contract IT companies. This keeps high tech high paying jobs and the revenue they generate within the local economy. In a closed source company, a large portion of the revenue goes back to the company and their local economy.

Linux is a good example. The software is free and documentation is online. There are free user support forums to give and receive help from other users who may have already solved an issue. If professional help is required, including training, this costs money, but it's generally quite reasonable and is fairly close at hand. These technicians and trainers do not generally work of a large multinational company, but are local to the community they serve. The dollars spent for supporting a company's users travel back to the local economy strengthening it.

With Microsoft Windows, very little of the cost of the software goes to the user's local economy. The lion's share goes to Microsoft which pays executive salaries, sales commissions, and wages of the production crew as well as material costs of the packaging of their product. Most of this revenue stays in the Redmond Washington area with some going overseas where the product is packaged. In practice this is a drain on the user's business and local economy while boosting an already thriving one. This is not a very logical or sustainable choice, especially in an economically depressed region.

Recently there has been a change in the marketing tactics used by nearly all major closed source software companies. I discovered this while trying to price equivalent closed source software to estimate the value of Open Source solutions. Many companies are offering free trials of their software giving you 30, 60, even 90 days to use a full version, while obscuring the price list. Initially I was simply annoyed at the lack of pricing data within a couple of clicks, but then I realized what was being done. While this strategy seems generous, it is not. If one uses the product for 90 days, the learning curve is largely over. At this point going back or switching to another solution simply repeats the learning curve and is counterproductive. The only other legal solution is to purchase the software at the end of the trial period.

What managers need to know when evaluating software is what the end cost is going to be. Business call this TCO, Total Cost of Ownership. How can effective management make a decision that affects their business productivity and budget if they do not know the cost of something before its use? They can't. Business, especially startups in an economy that kills half of the businesses that start within the first four years (Headd, 2002), cannot afford to give vendors a blank check like that. Ironically, when you finally do get the price for the software from these companies, all but two have increased by 10% or more in the two years since this survey was last done, in one case nearly doubling in price.

Flexibility and Freedom

In a business context, flexibility and freedom means the ability to choose the right tool for the job. Where this is selecting one tool from one vendor and one from another, this should be possible without prejudice. A business should be able to take a tool and change its use as the business needs evolve. More important in a large organization with its own IT staff is the ability to take a tool and modify it, streamlining its use within the company. Open Source software lets you do exactly that, leveraging the savings of seconds or minutes by multiplying it over the entire organization.

Standards compliance allows the interchange of data between applications. "To obtain flexibility at the architectural level, experience shows that it is often best to pick tried and trusted standards for interworking. If that is done, then best-of breed solutions can be selected for particular components within the architecture. Provided that the solutions can interwork suitably, the business should be able to avoid lock-in to a particular supplier and over-dependency." (GB Direct, 2006) Open source software by nature must be standards compliant to be useful, while vendor lock-in in closed source proprietary software allows a vendor to orphan the previous version in order to boost sales of the latest version of their software. It is not always possible to deploy a mixed version topology of the same software in an organization and keep the output files compatible without additional work.

The flexibility of Open Source offerings allow the tools to better match growth within a dynamic business environment. If a company doubles its staff, no additional licenses are required for Open Source software. Free is still free no matter how you multiply it. Even server based offerings have no limits to the number of seats, other than the capacity of the hosting hardware.

This freedom extends to modifying the program to better match the organization's needs. If an organization has a talented programmer on staff, or is willing to contract services to this effect, Open Source software may be modified with impunity. The only caveats to this are the implied responsibilities of the organization sponsoring the modifications.

The first responsibility is support. Suitability and stability of a modified program becomes the responsibility of the person or organization which modified it. It cannot be expected that the original developer support modifications to their code. Essentially, the modifying entity becomes the developer for the modified code base or fork. The second responsibility, in keeping with the terms of the applicable public licenses, the source code may need to be published in some manner to share the wealth and improve the product.

Support and Accountability

Accountability boils down to two trains of thought. First, who do I call if something goes wrong? This is your support network. Support of Open Source software is a mixed bag and generally falls under the heading of "what do you need?" For a private individual, this support generally comes from friends or relatives. A small business will generally hire a contractor to take care of a finite list of concerns. Big business will probably need its own IT staff anyway. Some of these will likely be talented programmers who can support the company's use of the company's standard software loadout.

All of these options have the option of company support and training for a modest fee as well as free options like support forums. This adaptive, whole package approach generally provides better support than someone in another country reading from a flip chart and ensures that your support staff is as invested in solving these issues as you are.

The second train of thought relating to accountability is "who do I sue when something goes wrong?" The short answer is no one. Software of all types generally have clauses which address liability in their license or End User License Agreement (EULA). The thought that one can sue Microsoft for billions of lost revenue may make someone feel more secure, but a quick read of the EULA that must be agreed to in order to install the software dispels this litigious fantasy of security through accountability. (Microsoft, 2005)

Why Companies Produce Open Source Software

Given the limited returns of Open Source software, one might wonder why a commercial company would produce these products. Open Source software is usually generated to fill a need, to scratch an itch as it were. Open Source business software is developed:

1. to provide an in house solution where none existed
2. to produce a solution for a one time cost where the cost of standard software was prohibitive
3. to use as a salvo to unseat a rival from a defacto monopoly
4. to produce a revenue stream by providing support or hosting for an online solution.

Oracle's Open Office is a clear example of this, but to fully understand, we have to learn some history. About the time Microsoft began to offer Windows, they diversified into office suites. By the mid to late 90's, Microsoft Office reigned supreme. If you did word processing, you used Word. If you did anything on a spreadsheet, it was done on Excel. To this day, the product names from the Microsoft Office suite are synonymous with the computing tasks they represent. For example, a "powerpoint" is a slideshow presentation, whether or not it was created on Microsoft's Powerpoint.

A little German company, StarDivision Gmbh, led by its then sixteen year old founder set about to create an office suite in 1984. Their product, StarOffice hit the market in various forms around 1990. Later, after the company had made it's money and had a good base of service clients to pay the bills, they offered the software for free. One year later in 1999, Sun Microsystems bought StarDivision lock stock and barrel for 73.5 Million dollars. (CNET, 1999)
This was a simple matter of economy for Sun. "The number one reason why Sun bought StarDivision

in 1999 was because, at the time, Sun had something approaching forty-two thousand employees. Pretty much every one of them had to have both a Unix workstation and a Windows laptop. And it was cheaper to go buy a company that could make a Solaris and Linux desktop productivity suite than it was to buy forty-two thousand licenses from Microsoft" (Phipps, 1999). This was not the only reason for the purchase though.

In 2000, Sun released the source code, and renamed the product Open Office in recognition of the fact that it now followed the open source model. It did not hurt Sun's feelings one bit that this product was a free alternative and direct competition to Microsoft's commercial offering, even improving the compatibility with Microsoft Office files in subsequent versions. Finally, when Oracle bought Sun Microsystems in January 2010, the only updates they made to the software or business model was to put the Oracle logo on the install and splash screens.

Learning for the success of Open Office and following the same model, Sun and later Oracle released the MySQL database, Java programming language, Glassfish application server, NetBeans development environment, OpenSolaris operating system, and the VirtualBox Virtual machine platform free for the taking with numerous other projects in the works (Oracle, 2010).

Conclusion

It is amazing what a group of dedicated software developers can accomplish, but the real quantum leaps happen when business and or government partner with the Open Source developer. These odd bedfellows bring disparate skills to the table, the strengths of one augmenting the weaknesses of the other.

The result is a software offering with security, transparency, flexibility, and auditability beyond the capabilities of commercial offerings. The TCO simply cannot be beat since the cost is front loaded with little to no repeat purchases. This saves some money in the first year, but the consecutive years will only reflect the labor costs of version upgrades and hardware improvements as the business grows.

These labor costs reflect the pay of IT staff and directly feed other local businesses, boosting local economies. Open Source software is the ultimate "win-win" scenario.

Apendix A: References

Business Dictionary (2010). *Software reliability deffinition.* Retrieved from http://www.businessdictionary.com/definition/software-reliability.html

Flink, J. J. (1977). *The car culture*. Boston: MIT Press.

GBdirect Ltd (2006). *Benefits of using open source software*. Retrieved from http://open-source.gbdirect.co.uk/migration/benefit.html

Headd, B. (2002). *Redefining business success: distinguishing between closure and failure* [Small Business Economics 21: 51–61, 2003]. (Report), Retrieved from http://www.sba.gov/advo/stats/bh_sbe03.pdf doi: Kluwer Academic Publishers

Himanen, P. (2001). *The Hacker ethic and the spirit of the information age*. New York: Random House.

Microsoft (2005). *Microsoft software license terms windows vista home basic windows vista home premium windows vista ultimate* [Section 25]. (EULA), Retrieved from http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBQQFjAA&url=http%3A%2F%2Fdownload.microsoft.com%2Fdocuments%2Fuseterms%2Fwindows%2520vista_home%2520premium_english_d16c019b-fa71-4fc9-a51d-a0621bddb153.pdf&ei=7uYKTLezIYf2MpelnLYE&usg=AFQjCNHWnXYPVXAXHkvBWrBcFy-HExFqSA

NSA (2009, January 15). *Security-enhanced linux*. Retrieved from http://www.nsa.gov/research/selinux/index.shtml

Oracle (2010). *Free and open source software*. Retrieved from http://oss.oracle.com/

OSI (2010). *Licenses by name.* Retrieved from http://www.opensource.org/licenses/alphabetical

OSI (2010). *The Open source definition.* Retrieved from http://www.opensource.org/docs/osd

Phipps, S. (Guest). (2006). *Lug radio, season 3 episode 10*. [Podcast]. Retrieved from http://www.lugradio.org/files/45/mp3-high/

Raymond, E. (2001). *The Cathedral and the bazaar*. Cambridge: O'Reiley.

Shankland, S. (1999, November 9). *Sun shelled out $73.5 million for star division*. Retrieved from http://news.cnet.com/Sun-shelled-out-73.5-million-for-Star-Division/2100-1001_3-232561.html

Appendix B: Listing of Open Source Software

This listing is a partial list of commonly used open source software. It is by no means complete or even exhaustive. The list replaces over $100,000.00 in closed source software

Audio and Video Players		
audacity	http://audacity.sourceforge.net/	Windows Media Player
Vlc	http://www.videolan.org/	Windows Media Player
Miro	http://www.getmiro.com/	Windows Media Player
Business and Management		
vtigerCRM	http://www.vtiger.com/	MS Dynamics CRM
xTuple	http://www.xtuple.com/	MS Dynamics CRM
OpenOffice	http://www.openoffice.org/	MS Office Ultimate
ProcessMaker	http://www.processmaker.com/open-source/	FileMaker Pro 11 Advanced
OpenWorkbench	http://www.openworkbench.org/	Microsoft Project Professional
OpenBravo	http://www.openbravo.com/	Quickbooks Enterprise Solutions
Gnucash	http://www.gnucash.org/	Quicken
ThinkingRock	http://www.trgtd.com.au/	Things (mac)
Graphics & Design		
BRL CAD	http://brlcad.org/	AutoCAD 2011
GIMP	http://www.gimp.org/	Photoshop CS5
Inkscape	http://www.inkscape.org/	Illustrator CS5
Blender	http://www.blender.org/	Autodesk Maya
Dia	http://dia-installer.de/	MS Visio
Kompozer	http://www.kompozer.net/	Dreamweaver CS5
Scribus	http://www.scribus.net/	Adobe InDesign
Wings3D	http://www.wings3d.com	Adobe AfterEffects
Cornice	http://wxglade.sourceforge.net/extra/cornice.html	ACDSee
Internet Applications		
Firefox	http://www.mozilla.com/en-US/	Internet Explorer
Thunderbird	http://www.mozillamessaging.com/	Outlook Express
Pidgin	http://www.pidgin.im/	AOL, Yahoo, MSN Messenger
RssOwl	http://www.rssowl.org/	FeedDemon, others
Filezilla	http://filezilla-project.org/	WS_FTP, CuteFTP, others
Server and Administation		
SME Server	http://wiki.contribs.org/Main_Page	Windows Home Server
Ubuntu Server	http://www.ubuntu.com/server	Windows Server 2008
Ubuntu Desktop	http://www.ubuntu.com/desktop	Windows 7 Ultimate
Ubuntu Netbook	http://www.ubuntu.com/netbook	Windows 7 Netbook
Apache	http://httpd.apache.org/	Windows Web Server 2008
Pandora FMS	http://pandorafms.org/	HP OpenView
OpenNMS	http://www.opennms.org/wiki/Main_Page	HP OpenView
Zenoss	http://www.zenoss.com/	HP OpenView
NeDi	http://www.nedi.ch/	HP OpenView
Nagios	http://www.nagios.org/	HP OpenView
Open-Xchange	http://www.open-xchange.com/	MS Exchange Server
Alfresco	http://www.alfresco.com/	MS Sharepoint Server
OpenVPN	http://openvpn.net/	Check Point VPN
Wireshark	http://www.wireshark.org/	Commview
Endian Community Firewall	http://www.endian.com/en/community/overview/	MS Forefront TMG
Peazip	http://peazip.sourceforge.net/index.html	Winzip
Cucku	http://www.cucku.com/index.aspx	Arkeia Backup
Bacula	http://www.bacula.org/en/	Arkeia Backup
Amanda	http://amanda.zmanda.com/	Arkeia Backup

Appendix C: GNU General Public License version 3 (GPLv3)

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the

stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate

does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the

recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.
THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.
IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.
If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

> <one line to give the program's name and a brief idea of what it does.>
> Copyright (C) <year> <name of author>
>
> This program is free software: you can redistribute it and/or modify
> it under the terms of the GNU General Public License as published by
> the Free Software Foundation, either version 3 of the License, or
> (at your option) any later version.
>
> This program is distributed in the hope that it will be useful,
> but WITHOUT ANY WARRANTY; without even the implied warranty of
> MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
> GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License
> along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

> <program> Copyright (C) <year> <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.