
Open-source software

Lotte Mygind, 2009

www.mjolner.dk

© 2009



Contents

- Introduction
- Copyright and software licenses
- What is open-source software?
- History of open source
- Common open-source software licenses
- Open source in business
- Embedded systems
- Practical advice
- Conclusion



My background

- Employed in Mjølner Informatics A/S
 - Software developer
 - Project manager
 - Advisor, e.g with respect to open source
- Started working with open source in an earlier job as product manager for a Linux distribution (2004-2007)
- Responsible for all open source processes in large international corporation



My view of open-source

- Open-source can speed up software development and cut costs
- There is a lot of excellent open-source software – e.g. development tools, web servers, browsers, operating systems, etc
- Not all open-source software is good
- It is possible to make money on open-source software but it requires some new strategies



D-Link – famous case

- D-Link is a Taiwanese vendor of networking solutions
- D-Link consumer products: Routers, network adapters, network storage, etc
- In 2005(?) D-Link launched their new wireless network storage: DSM-G600
- A 26-year old German, Harald Welte, bought the device and reversed engineered its firmware to find that it was Linux-based



D-Link – famous case

- Harald Welte contacted D-Link and told them that they did not fulfill the license terms of Linux
- “Do as the license prescribes and pay my expenses”
- D-Link refused to settle
- In Sept 2006, D-Link lost the case – ruled by a German court

What went wrong?

Why was this so important for D-Link that they refused to settle?



Copyright and software licenses

www.mjolner.dk

© 2009



What is copyright? ©

Copyright protects “original works of authorship” including source code. The owner of the copyright can authorize others to do the following:

- **To reproduce** the work;
- To prepare **derivative works**;
- **To distribute copies** of the work to the public by sale or other transfer of ownership or by rental lease or lending;

Without authorization from the author you are not allowed to use copyrighted source code.



What is a software license?

A software license grants a person (or company) a right to use software on particular terms.

The copyright takes away your rights to use the software.

The license gives you new rights to use the software.

You always have to read the license!



Patents

- It is possible in some jurisdictions to get a patent for software invention, e.g. an algorithm.
- Lately – great increase in the number of patents on software inventions.
- Commercial organisations often “bargain” patents.
- Many big, international software companies make a lot of money on patents!



Infringements

- Copyright infringement:
 - Copying of source code without permission
 - Not fulfilling license terms
 - Changing the license without permission
 - Using software that infringes on copyright
- Patent infringement:
 - Using someone else's patent without permission



Infringements in open source?

- Patent/copyright infringements in open source?
 - Anyone can read the source and look for infringements
 - Code written by many people in many organisations
 - Many open source programmers don't care!
- Microsoft claims that the Linux kernel and other mainstream open source software infringe on a number of their patents.



What is open-source software?



What is open-source software?

- “Open source” refers to software distributed in source form which can be freely modified and redistributed.
- Formal definition – see opensource.org
- Open-source software is **not** necessarily free of cost.
- Software that is released with the source code is **not** necessarily open source.



Open source definition

- 1. Free Redistribution:** The license shall not restrict any party from selling or giving away the software [...] 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. [...] well-publicized means of obtaining the source code for no more than a reasonable reproduction cost [...]



Open source definition, cont.

- 3. Derived Works:** The license must allow modifications and derived works, and must allow them to be distributed under the same terms [...]
- 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 [...]
- 5. No Discrimination Against Persons or Groups:** [...]
- 6. No Discrimination Against Fields of Endeavor:** [...]
- 7. Distribution of License:** The rights attached to the program must apply to all to whom the program is redistributed [...]



Open source definition, cont.

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 [...]

9. License Must Not Restrict Other 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: [...]



History of open source



History of open source

- The early days: Collaborative software development not formalized.
- 1983: Richard Stallman starts the GNU project.
- 1985: Stallman forms the Free Software Foundation (FSF), a sociopolitical organization that promotes “free” software.
- 1991: Linus Torvalds invites people to contribute to a free operating system and makes the first release of the Linux kernel software.



History of open source, cont.

- 1998: The term *open source* is coined by the *Open Source Initiative*.
- Today: Many vendors provide support for open-source software, such as Linux distributions, which typically bundle the Linux kernel with GNU programs and a graphical interface.





The idea of “free software”

- The FSF has created the GPL and LGPL licenses.
- The aim is to give *all* users the freedom to redistribute and change the software. ***Anyone who redistributes the software with or without changes must pass along the freedom to further copy and change it.***
- The freedom of the software must be protected – free software and derivative works must stay free.
- The FSF believes free software is *morally* and *technically* superior.



Technically superior?

- “With a thousand eyeballs, all bugs are shallow”
- But just because a project is open source, there will not necessarily be a thousand eyeballs...



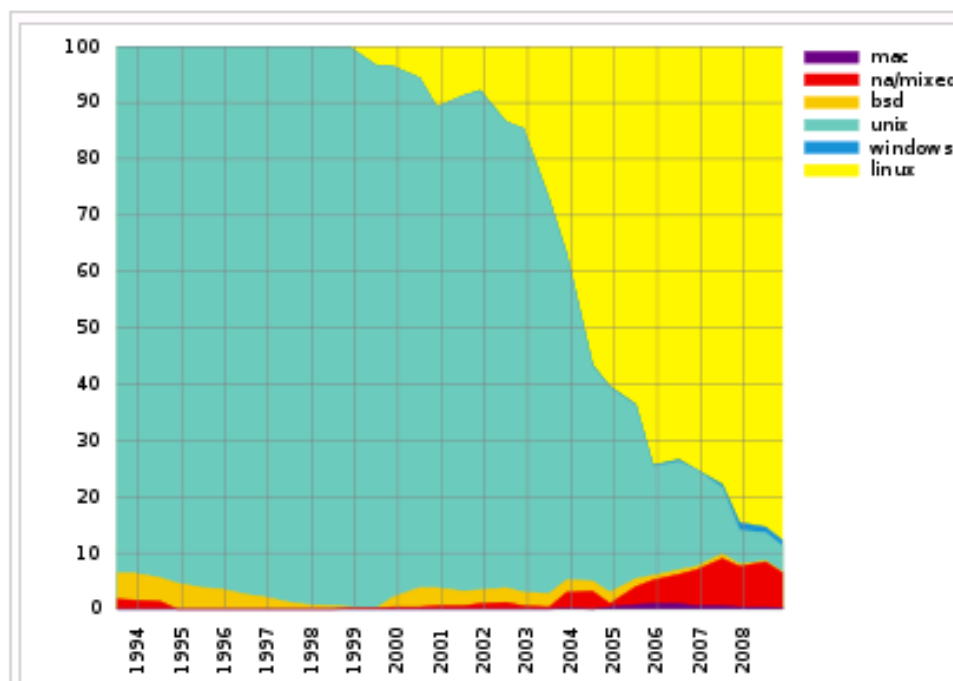
Open source statistics

- Apache web server is used by more than 60% of all web servers.
- Linux is used on a significant share of all internet servers.
- 150,000 projects on sourceforge.org and 7-10% are "active" (2007)
- GNU/Linux systems and NetBSD support a broader range of hardware platforms than any other operating system.



Open source statistics

- The fastest computer in the world, IBM Roadrunner, is Linux-based
- Operating systems for supercomputers:



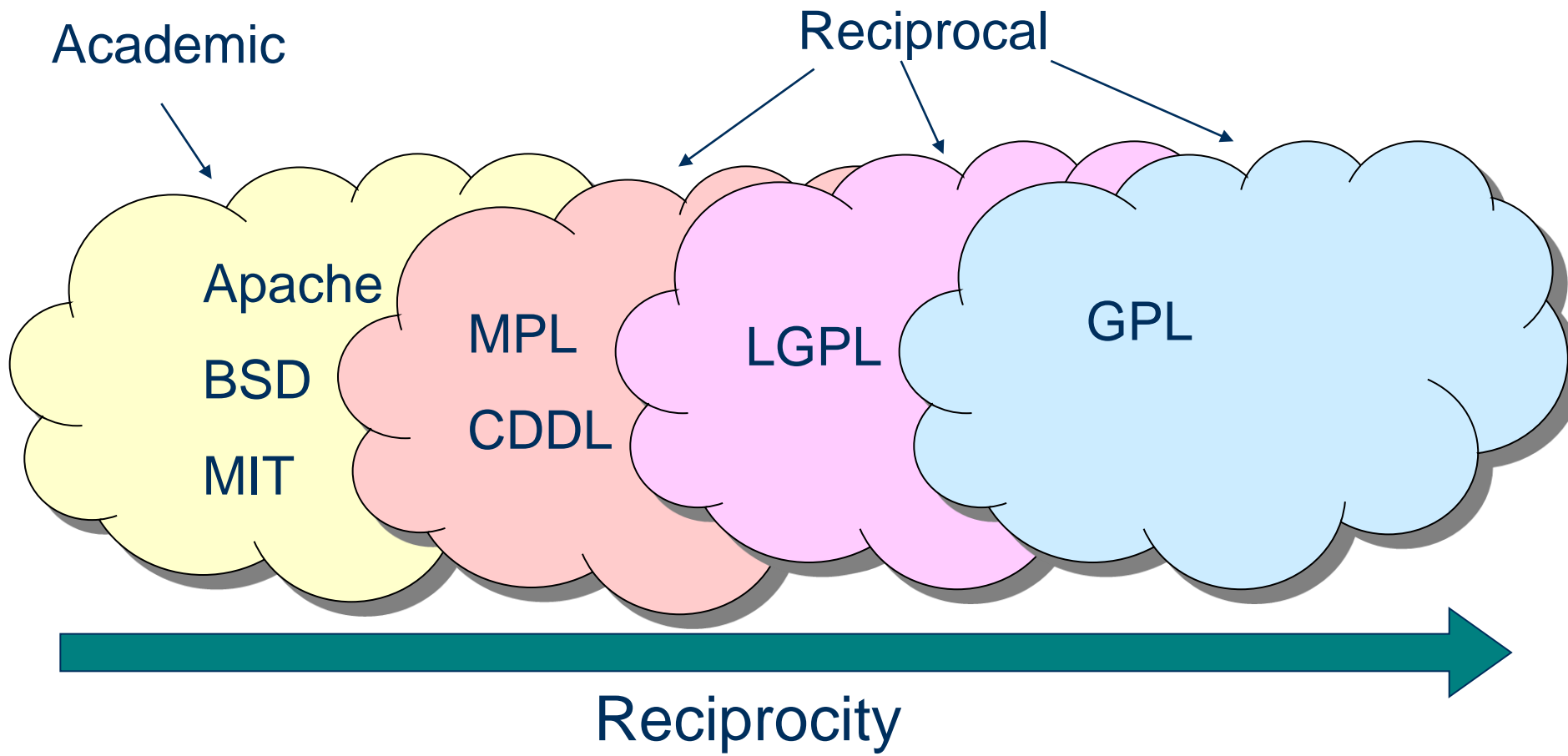
Common open-source software licenses

www.mjolner.dk

© 2009



Examples of licenses



Academic vs reciprocal

- An academic license allows the software to be included in *proprietary* software
- A reciprocal license requires that derived works are made open source under the same license
- But "derived work" is defined differently by the reciprocal licenses



GPL – GNU General Public License

- Reciprocal: Derived works must open sourced under the GPL
- So what is a derived work?
 - Defined by copyright law, but not tested very much in court
 - Dynamic linking is a grey zone
- The GPL is a *strong* reciprocal license



GPL version 3

- GPLv3 was released in June 2007
- GPLv2 is the dominant GPL license
- More aggressive with respect to software patents than GPLv2
- Efforts to make the license compatible with other licenses such as Apache
- Avoid "tivoization", i.e. the creation of a system in which the hardware prevents users from running modified versions of the software on that hardware
- Will GPLv3 be commonly used?



LGPL - GNU Lesser General Public License

- The LGPL is intended for libraries
- Reciprocal: Derived works must be open sourced under LGPL (or GPL)
- The definition of a derived work does not include programs linking with the library
- The license is very complex – so if you use, modify or link with LGPL, check the license or ask someone!



Other reciprocal licenses

- Common Development and Distribution License (CDDL) and Mozilla Public License (MPL)
- Reciprocal: Derived works must open sourced under the same license
- The reciprocity is only on *file basis*



Academic licenses

- Well-known examples: BSD, MIT, Apache
- Note that these licenses have older versions, and some of these have annoying “advertising clauses”
- The Apache has a “patent termination” condition



BSD

Copyright (c) <YEAR>, <OWNER>

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the <ORGANIZATION> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES ...



Patent retaliation

- Some licenses will terminate your rights if you assert legal claims against contributors, users or anyone
- Example – MPL:
 - lose contributor’s patent and copyright grants if you assert claim against contributor for work
 - lose Contributor’s patent grants if assert patent claim against any “software, hardware, or device”



Open source in business



How to make money on open source?

- Open source does not have to be free
- But once you have sold the software, you cannot prevent the customer to copy it and give it away for free
- Cannot charge a per-copy license fee



SUSE (Novell)

- SUSE was founded in 1992 in Germany
- First Linux distro in 1996
- Proprietary tool YAST (became GPL in 2004)
- Price for SUSE Linux Enterprise Server is 290-1240 euros
- You pay for support, documentation, proprietary tools
- Acquired by Novell in 2003 for \$210 millions



Other big actors

- Redhat (Linux)
- SUN (Java, OpenOffice.org)
- IBM (Eclipse, patent donations, m.m)
- MySQL AB (MySQL)



Business trends within open source

- Use of open source in software products increased dramatically within the last 5-10 yrs
- Many companies sell support of open source
- Some companies offer legal indemnification at an additional price
- Some companies "donate" patents to open source
- Open source is a way to influence standardization or a market



Business trends within open source, cont.

- Many companies that develop or support open source upstream all their developments
- Some big companies run their own open source projects
- Some open source developers get paid for their open source work
- There are many small companies (1-5 persons) that work with open source in an international market
- The public sectors in Europe are very interested in open source



Embedded systems



Embedded

- An **embedded system** is a computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts.
- We consider **high-volume**, embedded **consumer** products



Characteristics

- Small things with high volume: Low cost per piece is essential
- Hardware must be low-cost
- High performance, low footprint, etc, are essential
- Open-source software is used more often than not in embedded products



Challenges

- Embedded consumer products can be reverse engineered by the consumer
- Strict resource constraints
 - > difficult to separate the application from the OS
 - > the application may become a derived work



Some GPL-cases and settlements

- D-Link
- Sitecom
- ASUS
- TomTom
- LIDL
- Note:
 - Smaller actors.
 - Many embedded products.
 - A lot of network equipment.



D-Link

- Linux has excellent support for networking and security – so Linux is often used by manufacturers of networking equipment
- D-Link probably had some code that they didn't want to share
- It may have been hard to separate their own software from the Linux kernel
- Did they think it wouldn't be discovered?
- Did management not know that the product included Linux code?



Practical advice



Advice for developers

- Deliver the source code of the open-source software unless you have a reason not to do so.
- If you do not deliver the source, read the license to check whether this is OK and whether there are new requirements in this case.
- Deliver the source code together with the end product.
- Never remove disclaimers, attribution notices, copyright, etc. Do not remove the license text unless you are certain that this is OK.
- Seek advice if you do not understand the license terms
- Keep your boss informed



For managers

- Do not be afraid to use open-source tools
- Educate developers so they know that you cannot just “use code that you find on the internet”
- Define a process of how to track usage of open-source components
- Consider to use a code investigation tool to find stuff that you didn’t know that you were using



Legal worries for managers

- You may face legal problems if you use open-source software with copyright or patent infringements
- If you do not fulfill the license terms, you may be sued, but - most likely - people will ask you to fix the problem in a nice way before suing you.
- If you have used software under a reciprocal license, you may be forced to make your own stuff open source
 - The developers did not know that GPL code might cause problems
 - The GPL code was used in a wrong way



Open source tools

- Open source software can be used for tools internally in your organisation
- Examples:
 - Use Linux as operating system
 - Use GIMP instead of Photoshop
 - Use Dia instead of Visio
 - Use OpenOffice instead of Microsoft Office
- The risks are almost non-existing



Open source in consumer products

- Legal risks:
 - Copyright infringements in code
 - Copyright infringements if the license is not fulfilled
 - Patent infringements in code
 - Can you defend your own organisation's patents?
- Support?
- Quality?
- Maintenance of your own modifications?



Same problems with proprietary software

- You may also be sued for patent or copyright infringements in software developed in your own organisation.
- ... or sourced from a vendor.



Conclusion



I hope you can answer...

- Is open source software free of charge?
- Who should I give the source code to?
- I found some source code on the Internet – is it open source?
- Is it risky to use open source?

