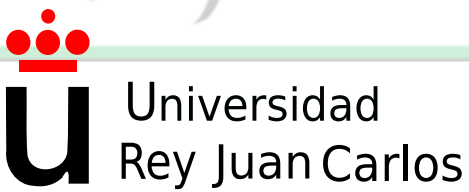
Legal Issues on FLOSS

Master on Libre Software

Gregorio Robles, Miquel Vidal

{grex,mvidal}@gsyc.es
GSyC/Libresoft – URJC

28 de noviembre de 2008



Contents

Intellectual Property (IP)

IP: Introduction (1/2)

Author's rights

- The type of works considered include: literature works (novels, poems, theater works, reference documents, newspapers and software), artistic works, scientific works, databases, movies, musical compositions and choreographies, architectonic works, publicity, maps and technical paintings.
- The protection is given to the form, the continent, the expression of the creative idea, but not on its contents.
- Neither the material for inspiraction (facts, dates...) nor the ideas are protected.
- In software the expression is given by the code; algorithms are not protected.
- Neighboring rights.

IP: Introduction (2/2)

Differences between the "common law" (where IP groups everything) and the "Continental law" (where IP and Industrial Property exist as different issues) system.

WIPO gives an "common law" definition of IP, but an approximation:

- *Intellectual property refers to creations of the mind: inventions, literary and artistic works, and symbols, names, images, and designs used in commerce.*

Origins

With the invention of the press, works became commercial objects.
First forms of plaguiarism appeared, so that editors forced legislators to regulate and protect the original works.
Regulation was also conceived as a way of controlling information (i.e. censorship)

Copyright on software (I)

Software came first as part of a hardware system (*bundling*)
The necessity of copyright began to arise when IBM had to split its business due to anti-monopoly legislation in the U.S.: software began to be distributed in an independent manner.
The goals of the copyright were:

- Protect investments in the development
- Promote distribution of works
- Protect the creative human activity by providing incentives
- Protect a technology very prone to be copied

At first, there was a big debate about if software should be protected by patents or by copyright.

Copyright on software (II)

Copyright was finally chosen because of following characteristics:

- Simplicity (no registration, no formalities...)
- Automatic
- Inexpensive
- No novelty, just originality (it may be state of the art!). Authors' right: personal and not trivial. Copyright: just personal.
- Includes documentation
- International (several conventions on copyright)
- Armonization with other works

Adapting the concept of copyright to software is not an easy task as there are many exceptions and special circumstances.

IP: Concepts

- IP: refers to several issues, depending on its context.
- Today, very often used referring to the privileges of non-physical goods with economic value.

Examples (Common law system):

- Copyrights: Protect from unauthorized copy: artistic or literary works, computer programs, data collections, industrial designs, etc.
- Trademarks: Protect company symbols and names.
- Trade secrets: Protect access to some industrial secrets.
- Patents: Protect the rights of exploiting inventions as monopolies.

Sometimes (for instance, in the Continental law system):

- Intellectual Property: only for copyright issues.
- Industrial Property: rest of issues (patents, trademarks...).

Copyright on software (and III)

What in software falls under copyright:

- The computer program (i.e. instructions, in any form): source code and object code!
- The description of the program (for instance, its UML)
- Additional material (user manuals, guides, etc.)
- Interfaces (graphics, sound, typographies...)
- Databases

Authors and copyright holders

- Authors is a (physical or juridical) person that creates a work.
- Collaborative work: unitary result of the collaboration of several authors where the input of each author may be identified and exploit independently.
- Collective work (art. 8 LPI): under the initiative and coordination of a physical or juridical person. It groups the input of several authors that cannot be identified independently and that compose a unique and autonomous creation. Examples: GNOME, Mozilla, FSF, etc.
- The work produced by an employee or by means of a contract is owned by the company. But the moral rights are still retained by the programmer.

The Copyright: Protection for Authors

The rights protected by copyright laws:

- Moral rights. Guarantee work dissemination and author attribution. Only in Continental law.
- Exploitation rights ("copyright" *per se* in Common law). Guarantee economic exploitation.

Exploitation rights have time expiration, depending on local laws. For example, in EU they expire 70 years after the author's death.

How can we grant some rights to users of copyrighted works?

Moral rights

- Disclosure of the work
- Way of publication: with his name, a pseudonym or anonym
- Recognition of his name and respect to his authors' status
- The right of the integrity of his work
- The withdrawal of his work (addressing compensation if needed)
- To access a unique exemplar if in possession of any other person

These rights cannot be withdrawn, cannot be transfered, are inalienable and some even perpetual.

Exploitation rights

- Reproduction (includes communication and copying): loading, presentation on the screen, execution, transmission and storage.
  - Even for using a program you require the author's aproval!
  - The right to copy/reproduce is fundamental in licenses; else the software cannot be run.
  - If somebody steals a book, he is attempting against the owner of the book but not the owner of the IP. If you copy a software without authorization, you are attempting against the owner of the IP as you need his approval to reproduce (copy) it.
- Distribution: Public disposal of physical copies (i.e. offering the software over the Internet is not included). Software is not sold as this could make re-selling possible. What is sold is the CD; the software is licensed!
- Public performance (there is no distribution of physical copies; What is public and private on the Internet?)
- Transformation (for instance, translation)

Licenses: Concepts

License:

- An unilateral "contract" between the author and the user.
- Grants some rights to the users of copyrighted work.

You don't need to sign the contract, but in that case, you don't have any rights over the copyrighted work. EULA is not necessary.

Piracy

With computers and Internet, the non-authorized copies received a name:

"Piracy"

Several organizations fight the "piracy". Example:

- Business Software Alliance (BSA).

This is a controversial word because it means "act of robbery over high seas" (with violence).

Trade secrets

- A trade secret is a way to protect investments in industrial area, through Industrial Property laws.
- Under trade secrets, there are several goods such as chemical or pharmaceutical formulas, but also software.
- Proprietary software enterprises hide the source code of their software products as a way to protect their investment in creating such software.
- One of the objectives, for example, is to avoid the creation of derivative works.
- However, in some countries, reverse engineering is permitted in order to create compatible software products.

Patents

- Patents?
  - The invention is not protected by secret. On the contrary, the invention is publicly available.
  - However, for a certain period (from 17 to 25 years) for exploiting the invention the interested company must pay a license.
- From other point of view, the patent is not a right to use the invention, but provides the right to exclude others from making, using, selling, offering for sale, or importing the patented; for the signed period.
- Patents, good or bad?

Software patents

Some comments about software patents:

- In some countries they are not legal.
- For example, Europe does not accept software patents, but in USA.
- Although not legal, in practice, lots of algorithms, and in fact, ideas, have been patented.
- Since trivial ideas (implemented with algorithms) are patented, they are often used by owners to drown competitors.
- "Patent trolls": Companies with a "patent portfolio", which sues other competitors for infringement of patents while doing little to really innovate.
- It is very easy to infringe a lot of patents when developing a software project.

Trademarks

- A trade name, is the name which a business trades under for commercial purposes, although its registered, legal name, used for contracts and other formal situations, may be another.
- Trading names are sometimes registered as trademarks or are regarded as brands.
- In Free Software, there are not very important, probably because registering a trademark is not free and most developers do not pay attention on them. However, there are some well known trademarks in this world, such as GNOME, GNU, Debian.
- Sometimes, the names are not registered in most countries and this implied some problems. For example, in USA somebody registered the trademark “Linux” and tried to obtain money for its use.

References

- Text for CC Attribution-ShareAlike 3.0 License
  `http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode`
- Debian Free Software Guidelines
  `http://www.debian.org/social_contract`
- The Four Freedoms (FSF-Europe)
  `http://www.fsfeurope.org/documents/freesoftware.en.html`
- FSF recopilation on FLOSS licenses
  `http://www.gnu.org/licenses/`

Introduction to Legal Aspects of Libre Software

The importance of licenses

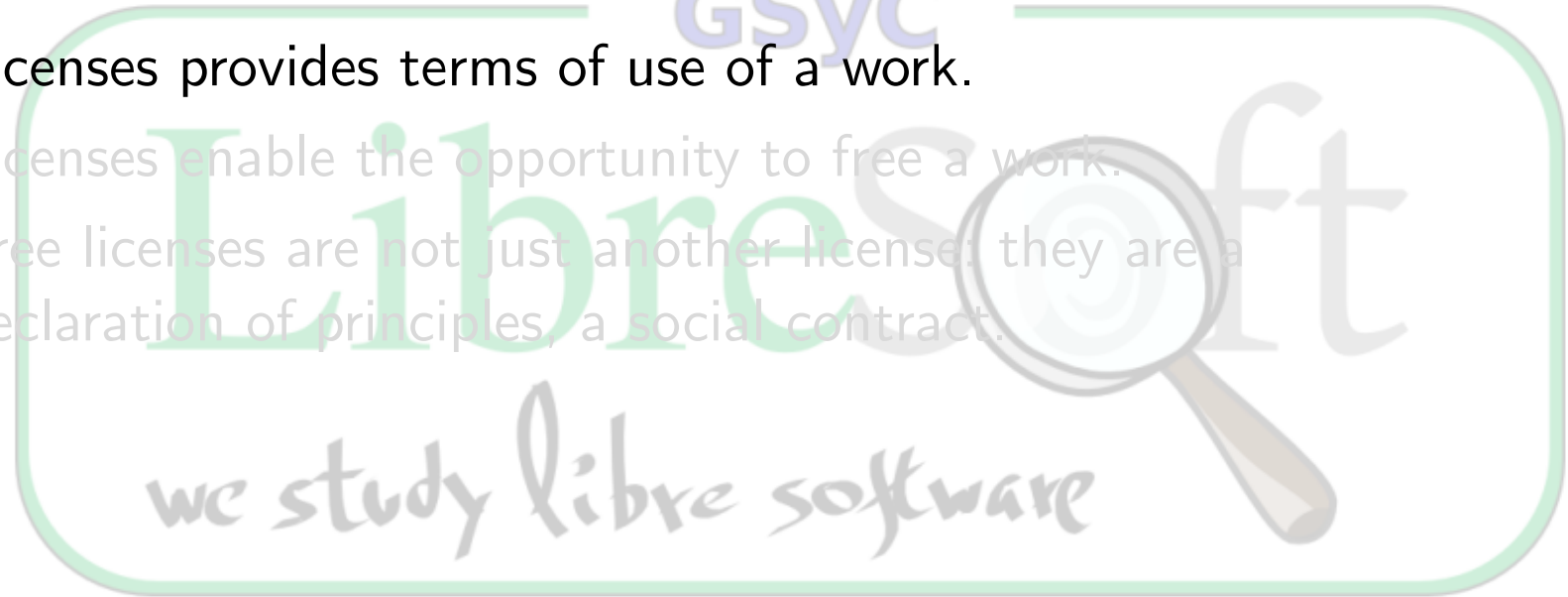
Why should we care about licenses?

- Licenses provides terms of use of a work.
- Licenses enable the opportunity to free a work.
- Free licenses are not just another license, they are a declaration of principles, a social contract.

The importance of licenses

Why should we care about licenses?

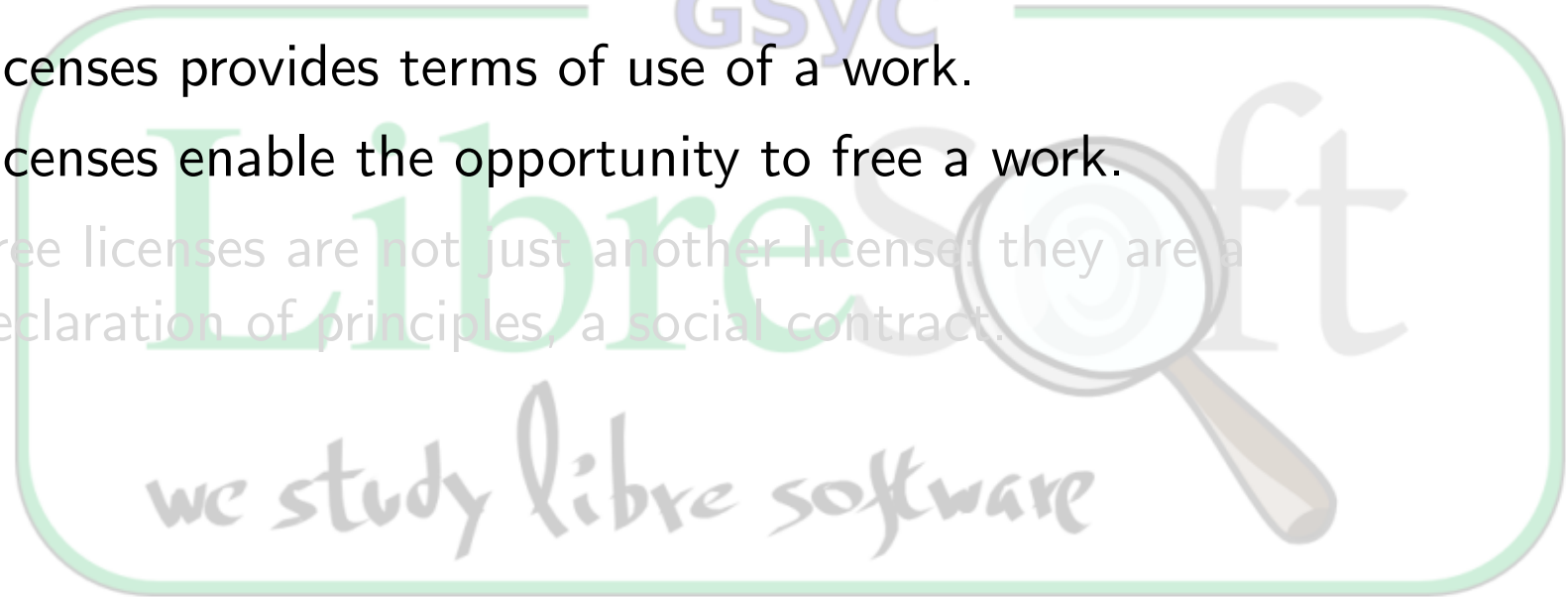
- Licenses provides terms of use of a work.
- Licenses enable the opportunity to free a work.
- Free licenses are not just another license: they are a declaration of principles, a social contract.



The importance of licenses

Why should we care about licenses?

- Licenses provides terms of use of a work.
- Licenses enable the opportunity to free a work.
- Free licenses are not just another license: they are a declaration of principles, a social contract.



The importance of licenses

Why should we care about licenses?

- Licenses provides terms of use of a work.
- Licenses enable the opportunity to free a work.
- Free licenses are not just another license: they are a declaration of principles, a social contract.

The importance of licenses

It's important to make a difference between two concepts:

- **Intellectual Property:** for the sake of clarity, it is better to label it copyright.
- **Industrial Property:** it refers to patents and trademarks and it is covered under different legislation.

Both categories are conceptually distinct, work in different ways, and are covered by a different piece of legislation.

In English-speaking countries (Common Law) industrial property is "philosophically" considered as part of intellectual property creating some confusion between both categories.

The importance of licenses

It's important to make a difference between two concepts:

- **Intellectual Property:** for the sake of clarity, it is better to label it copyright.
- **Industrial Property:** it refers to patents and trademarks and it is covered under different legislation.

Both categories are conceptually distinct, work in different ways, and are covered by a different piece of legislation.

In English-speaking countries (Common law) industrial property is "philosophically" considered as part of intellectual property creating some confusion between both categories.

The importance of licenses

It's important to make a difference between two concepts:

- **Intellectual Property:** for the sake of clarity, it is better to label it copyright.
- **Industrial Property:** it refers to patents and trademarks and it is covered under different legislation.

Both categories are conceptually distinct, work in different ways, and are covered by a different piece of legislation.

In English-speaking countries (Common law) industrial property is "philosophically" considered as part of intellectual property creating some confusion between both categories.

The Legal Framework for Free Licenses

- Free licenses are based on international copyright laws and provide the user with certain freedoms. These are granted as permissions which could not be exercised without the license (by default "all rights are reserved").
- Legal Hacking: free licenses behave as any other license except that they grant a number of rights to the user rather than restricting them (about meme "does this license apply in my country?").

Legally, the only difference between proprietary and libre software is the license (i.e. terms of use). Licenses (free or not) are based on every country's *copyright* law.

Example

Implement a basic free license is very easy:

Free License Example

Copyright (c) 2008 Foobar Developers. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the redistributions of source code must retain the above copyright notice.

That's all!!

The Libre Software Definition

Free licenses are based on libre software licenses. They are the mechanism to legally implement the four freedoms that the author or license-holder granted to users.

When you receive a libre software you get:

- Freedom 0 The freedom to use (run) the program, for any purpose.
- Freedom 1 The freedom to study how the program works, and adapt it to your needs.
- Freedom 2 The freedom to redistribute copies.
- Freedom 3 The freedom to improve the program, and release your improvements to the public, so that the whole community benefits.

Freedoms 1 and 3 require access to a source code. All four freedoms must be granted at the same time.

The Libre Software Definition

Free licenses are based on libre software licenses. They are the mechanism to legally implement the four freedoms that the author or license-holder granted to users.

When you receive a libre software you get:

- Freedom 0 The freedom to use (run) the program, for any purpose.
- Freedom 1 The freedom to study how the program works, and adapt it to your needs.
- Freedom 2 The freedom to redistribute copies.
- Freedom 3 The freedom to improve the program, and release your improvements to the public, so that the whole community benefits.

Freedoms 1 and 3 require access to a source code. All four freedoms must be granted at the same time.

Some concepts related with FLOSS licenses

Some concepts interesting for us:

- Use: The right to use (run) the program, for any or some purposes.
- Redistribution: The act of copying the program and giving it to others.
- Derivative work: A program based in other program, reusing its source code.
- Authorship attribution: The obligation of recognizing the authorship of a work when applying any change, such as deriving or redistributing it.

The property of the program never changes: the program is always owned by the license-holder. With the license, the user only get some rights of use.

Types of free licenses

Every free license, no matter the kind of work, must guarantee the four freedoms mentioned above for the case of software (use, copy, modification, and redistribution).

However, there are free software licenses more permissive and other more strict (the most strict licenses are known as "copyleft" licenses).

Please note that two non-compatible free licenses does not imply that one of them is "less free" than the other.

Types of free licenses

Every free license, no matter the kind of work, must guarantee the four freedoms mentioned above for the case of software (use, copy, modification, and redistribution).

However, there are free software licenses more permissive and other more strict (the most strict licenses are known as "copyleft" licenses).

Please note that two non-compatible free licenses does not imply that one of them is "less free" than the other.

Types of free licenses

Every free license, no matter the kind of work, must guarantee the four freedoms mentioned above for the case of software (use, copy, modification, and redistribution).

However, there are free software licenses more permissive and other more strict (the most strict licenses are known as "copyleft" licenses).

Please note that two non-compatible free licenses does not imply that one of them is "less free" than the other.

Types of free licenses (2)

Free licenses can be classified in two main categories:

- Copyleft licenses: The author retains copyright and permits redistribution and modification provided all such redistribution is licensed under the same license. Additions and modifications by others must also be licensed under the same 'copyleft' license. Also known as "reciprocal licenses" or "share-alike" (GPL, GFDL, CDDL, CC-by-sa).
- Permissive licenses: The author retains copyright solely to disclaim warranty and require proper attribution of modified works, but permits redistribution and modification in any work, even proprietary ones (CC-by, *BSD, Apache, MIT).

Please note that both license types are for "Libre software". But with the first type, you can do proprietary derivative works, and with the copyleft license not.

Types of free licenses (2)

Free licenses can be classified in two main categories:

- Copyleft licenses: The author retains copyright and permits redistribution and modification provided all such redistribution is licensed under the same license. Additions and modifications by others must also be licensed under the same 'copyleft' license. Also known as "reciprocal licenses" or "share-alike" (GPL, GFDL, CDDL, CC-by-sa).
- Permissive licenses: The author retains copyright solely to disclaim warranty and require proper attribution of modified works, but permits redistribution and modification in any work, even proprietary ones (CC-by, *BSD, Apache, MIT).

Please note that both license types are for "Libre software". But with the first type, you can do proprietary derivative works, and with the copyleft license not.

Types of free licenses (2)

Free licenses can be classified in two main categories:

- Copyleft licenses: The author retains copyright and permits redistribution and modification provided all such redistribution is licensed under the same license. Additions and modifications by others must also be licensed under the same 'copyleft' license. Also known as "reciprocal licenses" or "share-alike" (GPL, GFDL, CDDL, CC-by-sa).
- Permissive licenses: The author retains copyright solely to disclaim warranty and require proper attribution of modified works, but permits redistribution and modification in any work, even proprietary ones (CC-by, *BSD, Apache, MIT).

Please note that both license types are for "Libre software". But with the first type, you can do proprietary derivative works, and with the copyleft license not.

Types of free licenses (2)

Free licenses can be classified in two main categories:

- Copyleft licenses: The author retains copyright and permits redistribution and modification provided all such redistribution is licensed under the same license. Additions and modifications by others must also be licensed under the same 'copyleft' license. Also known as "reciprocal licenses" or "share-alike" (GPL, GFDL, CDDL, CC-by-sa).
- Permissive licenses: The author retains copyright solely to disclaim warranty and require proper attribution of modified works, but permits redistribution and modification in any work, even proprietary ones (CC-by, *BSD, Apache, MIT).

Please note that both license types are for "Libre software". But with the first type, you can do proprietary derivative works, and with the copyleft license not.

Permissible restrictions

- Attribution of authors (such attribution does not impede normal use of the work).
- Transmission of freedoms (copyleft or share-alike).
- Protection of freedoms (access to source code or prohibition of “technical measures”, DRM).

What is Copyleft?

- The FSF considered insufficient to grant the four freedoms mentioned above (use, copy, modification and redistribution).
- Copyleft makes sure that all users receiving a copy of the program get also the original four freedoms.
- It is an active defense of user's freedoms.
- The copyleft clause might have diverse implementations but all of them share the same concept: distribution of any version of this program must use this same license.

Dual-licensing

An license-holder can distribute his work under more than one license.

- License compatibility (Perl, Mozilla/Firefox, MySQL).
- Market segregation based business models (MySQL Enterprise)
- Allows the license-holder to offer customisations, early releases, generate other derivative works or grant rights to third parties to redistribute proprietary versions.

Since the author or license-holder is the owner of the program, he can always select the license of his program; for example, he can change from a free license to a proprietary one. However, the version with free license will ever exists and the conditions for this version will not change (forks such as GForge).

Conclusion

It is essential to pay attention to free licenses and learn how to differentiate among them, not only in order to determine if we are subject to any restriction –as users– or what options we are really giving the users –as authors–, but also to understand possible compatibility issues derived from combining contents covered by different free licenses.

Free Licenses

General Public License

What makes the GPL so special?

- It was the first license to outline the copyleft principle.
- All non-restrictive licenses have been based on the GPL, including Creative Commons and the Wikipedia license.
- Without the GPL, copyleft would be just an idea, and Free Culture would have not existed (as least as we know it).

Free licenses are the juridical-philosophical foundation of libre software

General Public License

What makes the GPL so special?

- It was the first license to outline the copyleft principle.
- All non-restrictive licenses have been based on the GPL, including Creative Commons and the Wikipedia license.
- Without the GPL, copyleft would be just an idea, and Free Culture would have not existed (as least as we know it).

Free licenses are the juridical-philosophical foundation of libre software

General Public License

What makes the GPL so special?

- It was the first license to outline the copyleft principle.
- All non-restrictive licenses have been based on the GPL, including Creative Commons and the Wikipedia license.
- Without the GPL, copyleft would be just an idea, and Free Culture would have not existed (as least as we know it).

Free licenses are the juridical-philosophical foundation of libre software

General Public License (2)

- Based in Emacs license, first copyleft license (1986).
- GPL version 1 (1989). Generics, program-independent, "version 1 or later".
- GPL version 2 (1991). "Liberty or dead" clause (it prevents from patents threats).
- GPL version 3 (2007. Tivoization, patents and DRMs. Community discuss.

GNU GPL License

GPL = GNU General Public License.

GPL Concepts:

- GPL = GNU General Public License.
- Created by FSF for the GNU Project.
- However, very often used in Free software non-GNU.
- Probably, the most popular Free Software license: around 70 % Freshmeat projects licensed under GPL.
- Some popular software licensed under GPL: Linux, GNOME, Emacs, GCC, Epiphany...

GNU GPL License (II)

GPL Characteristics:

- This license guarantees the four FLOSS freedoms.
- It is though to always guarantee code freedom. This is the meaning of the "copyleft" clause: all derivative works should be licensed also under the same license.
- Since in USA software patents are possible, GPL includes a clause for avoiding GPL licensing of patented software or algorithms.
- GPL has three versions. The most known is GPL Version 2 because has been on the market for more than 10 years. However, in 2006 the GPL V3 born.
- GPL code can not be mixed with other code under "GPL-incompatible" license.

GNU LGPL License

LGPL = Lesser GPL

- LGPL started as "Library GPL". Later renamed to "Lesser GPL".
- LGPL maintain all freedoms and restrictions to the licensed software, but with one exception:
  - LGPL Software can be integrated/linked with any other software, without limitations (including proprietary software).

Why LGPL?

- Created for promoting the use of GPL libraries in any other software (ex: GNU libc).
- Later, FSF checked that LGPL did not helped to the creation of new Free software, so they decided to rename it to "lesser" and discourage its use.

Other robust licenses

- **Sleepycat**. This license is used, for example, in Berkeley DB. Very similar to GPL (including copyleft clause), although shorter.
- **eCos**. It is used in eCos, a real-time OS. It is a GPL modification, in order to avoid the restriction of integration between GPL software and GPL-incompatible software. In fact, it is a LGPL.
- **Affero GPL**. It is though for programs which do web service. It includes a clause for mandatory maintaining the web redistribution of source code, if the author wanted it.
- **IBM Public License 1.0**. This license, similar to GPL, also includes the obligation of a free licensing of any patents used in any later modification.
- **Mozilla PL 1.1**. It is a robust FLOSS license created by Mozilla Foundation when decided to release the first FLOSS version of their navigator, based in old Netscape code.

Other robust licenses: Affero GPL

Affero GPL (AGPL) is a free software license, copyleft-like, published by the Free Software Foundation.

- It is a derived license from GPL.
- It contains a clause requiring distribution of any modified source code of applications running in a computer network.
- It aims to cover the case of modified GPL software which is not distributed because the GPL license does not require to do so (web services or online applications).

BSD Licenses

Origins:

- BSD (Berkeley Software Distribution) is a Unix flavor developed by University of Berkeley (CA).
- BSD Unix was licensed under a "minimalistic" license which permits both source or binary redistribution; also modifications, but without any other restriction.

The BSD licenses:

- Based in original BSD license.
- Very popular: X-Window, Tcl/Tk, etc.
- You may redistribute the work, in any form (source or binary) but with all remaining copyright notes (authorship attribution).
- There is a "no warranty" clause. Very criticized, this clause is for avoiding legal problems with the use of software and is also very usual in proprietary software.

A BSD license

Copyright (C) (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:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. 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.
3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

A BSD license: No Warranty

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Other permissive licenses

- **X-Window R11 License (X11)**. This license applies to the most used graphical subsystem in Unix systems. The license is very similar to BSD and is also known as "MIT license". Some other derivative works are licensed under X11 License, such as XFree86.
- **Zope Public License 2.0**. This license is used by the Zope distribution (an application server) and some related products. Near BSD license, also prohibits the use of Zope Corporation trademarks.
- **Apache**. Very similar to BSD, lies in the license used by almost all Apache project products.
- **Public domain**. An intellectual work in the public domain is neither under any IP law nor a license. Most public domain works retains, however, the authorship. For this reason, it is very similar to have the program under PD or under a BSD license.

Free licenses for other intellectual works

Free Licenses for other intellectual works

- FLOSS licenses have inspired licenses for other intellectual works: audio, video...
- Stallman distinguished between functional works (documentation, encyclopedias, manuals, etc.), and non-functional works (literature, music, movies, etc.) to justify the use of restrictive clauses for works other than software.
  - Is it true that non-functional works require less freedoms?
  - legitimation of non-commercial clauses (and other kinds).

Free Licenses for other intellectual works

- FLOSS licenses have inspired licenses for other intellectual works: audio, video...
- Stallman distinguished between functional works (documentation, encyclopedias, manuals, etc.), and non-functional works (literature, music, movies, etc.) to justify the use of restrictive clauses for works other than software.
  - Is it true that non-functional works require less freedoms?
  - legitimation of non-commercial clauses (and other kinds).

Free Licenses for other intellectual works

- FLOSS licenses have inspired licenses for other intellectual works: audio, video...
- Stallman distinguished between functional works (documentation, encyclopedias, manuals, etc.), and non-functional works (literature, music, movies, etc.) to justify the use of restrictive clauses for works other than software.
  - Is it true that non-functional works require less freedoms?
  - legitimation of non-commercial clauses (and other kinds).

"Freedom Defined": free cultural works

- It is not a new license, but an initiative to provide a definition of *free* outside of the software world to end the current ambiguity of the term in the Free Culture movement.
- It was a Debian developer proposal and it reached a consensus over a draft with the community, the FSF and CC.
- It is an adaptation of the four essential freedoms of Free Software.
- It is not about the author's right to decide the terms in which he wants to share his work, it is about getting knowledge of which licenses (and works) do or do not fit the definition of libre.

Identifying Free Cultural Works

Whenever the user of a work cannot legally or practically exercise his basic freedoms, the work cannot be considered and should not be called "free". So:

- Non-commercial restrictions are non-free licenses.
- No derivative restrictions are non-free licenses.
- NonCommercial Sharealike Is Not Copyleft: Copyleft is a reversal of copyright. It restores and protects the rights that copyright removes. The rights, not *some* rights.
- "Open Content" and "Open Access" are not a clear definition of freedom.

`http://freedomdefined.org/`

GNU FDL

GNU has the "Free Documentation License" (GFDL). In this license:

- There are distinction between "transparent" copies of the document (similar to "source") and "opaque" copies (similar to "binary").
- There are some additional restrictions: acknowledgments, dedications, and the history of the document can be modified but only by adding new lines.
- The document could include "invariant" and "cover" sections, not modifiable. However, only "non-technical" texts can be considered invariant.
- GFDL does not comply with Debian guidelines (DRM clause, GPL incompatible and invariant parts), unless the invariant section clauses are not used.

Creative Commons Licenses

Creative Commons objectives

Creative Commons, a non-profit organisation for...

- Creating a set of licenses, for any type of content.
- Indexing CC licensed works, so you can locate free contents fast...
- International adaptation of licenses (for example, there are Spanish official CC licenses)

http://www.creativecommons.org/

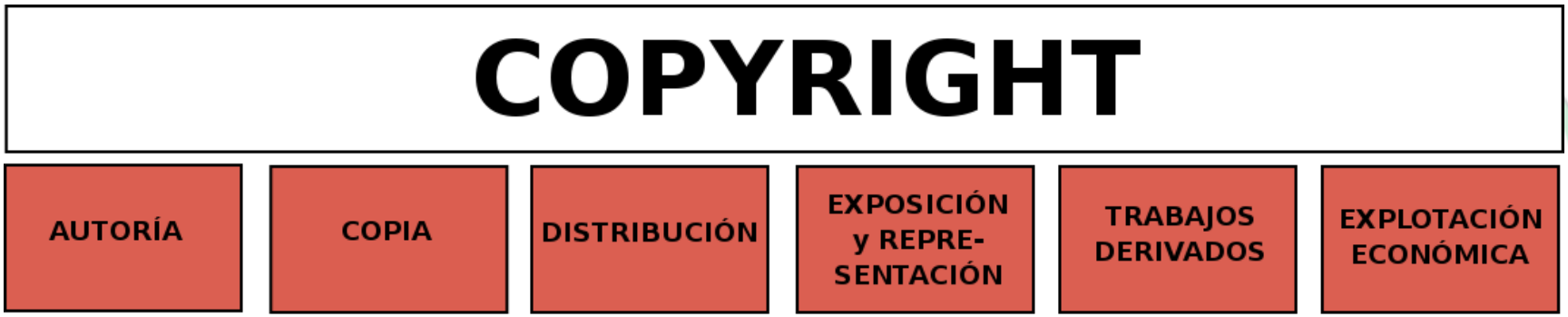
Creative Common Licenses objectives

With Creative Common licenses you can...

- Donate your work to the public domain or...
- Maintain some rights:
  - Attribution
  - Non-commercial
  - No derivative
  - Share Alike
- You can **combine** these rights (with some logical exceptions). So, you maintain

some rights reserved.

Copyright: All rights reserved



Public Domain: No rights reserved

Creative Commons: Some rights reserved

Types of Creative Commons licenses (1/2)

Mixing and matching these conditions produces six possible combinations:

CC Licenses

1. Attribution (by)
2. Attribution + Non commercial (by-nc)
3. Attribution + No Derivate Works (by-nd)
4. Attribution + share alike (by-sa)
5. Attribution + Non commercial + No Derivate Works (by-nc-nd)
6. Attribution + Non commercial + share alike (by-nc-sa)

Types of Creative Commons licenses (2/2)

"Special" CC licenses:

- Public Domain Dedication (PD)
- Founder's Copyright: the work is released into PD after 14 or 28 years.
- Sampling Plus: parts of the work can be copied and modified for any purpose. The entire work can be copied for non-commercial purposes.
- Noncommercial Sampling Plus: the whole work or parts of the work can be copied and modified for noncommercial purposes.

Retired licenses: Developing Nations, Sampling.
There are also some licenses intended for software, but it is recommended to use those available from the free / open source software world.

Use of Creative Commons licenses

All CC licenses contain three sections, intended for humans, lawyers and machines:

- Commons Deed: It is a summary, human readable, of the license with the relevant icons.
- Legal Code: The complete legal code in whch the chosen license is based.
- Digital Code. The digital code, a machine-readable version that helps search engines and other applications identify the work by its terms of use.

The Attribution-ShareAlike license

With this license, you reserve two rights:

- Attribution of your work.
- Any derived work must be licensed under same license.

There are not restrictions in commercial use or derivative works. So, this license is the nearest to a FLOSS license. However, there are some problems...

Debian-legal enters the arena

After the popularization of CC, Debian-legal entered and warned of Attribution. The causes were:

- A work under Attribution can be in fact, NoDerivs.
- Inaccurate or excessive authorship credits.
- Problems with Anti-DRM clause.
- Trademark restrictions.

Attribution[-ShareAlike] is not compatible with DFSG

Attribution Problems (I)

- Licensor/Original Author can request removal of him/her references in derived works.
- However the author name must be included in derivative works "if supplied" (Attribution)

It's unclear if creator of derivative works can comply with a requirement of removing references and requirement of give attribution. Then, a request to remove references can make impossible to comply with Attribution; so it may not be possible to make derivative/collective works under this license (DFSG 3 & 1 incompatible). **The work is not free?**

Attribution Problems (II): Authorship credits

- Requirements for crediting Licensor for his/her work: it's ambiguous
- So, we need the most pessimistic interpretation:
  - Required attribution for the licensor everywhere that authorship credit is given.

Example 1: *If a work is a collection of essays by different authors, with authorship credit given in the chapter titles, the Licensor's name would have to be listed for each chapter title, even if they did not contribute to it.*
Example 2: *If Alice writes her autobiography, and includes lyrics from Bob's song in one chapter, she must give him credit for the entire work: "The Autobiography of Alice, by Alice and Bob", or even "The Autobiography of Alice and Bob."*

Attribution Problems (III)

- Anti-DRM Clause: Distribution with measures to control access is not compatible with DFSG 1.

Example 1: Private distribution of work might be forbidden.
Example 2: Distribution in a server with control access (i.e. firewall) might be forbidden.

Attribution Problems (IV)

- Web page says: Using CC trademark or logos is strictly prohibited when it's used for other than indicating the work is licensed under CC.
- Altough CC Trademark not appears to be part of the CC licenses, this is only warned in "source code" of CC web page.
- Interpretation: This can prevent redistribution, modification ...

NoDerivs problems

Obviously, NoDerivs licenses are not compatible with Four Freedoms (nor DFSG)

NonCommercial problems

Obviously, NonCommercial licenses are not compatible with Four Freedoms (nor DFSG)

Debian recommends to authors...

- All software licensed under CC is not compatible with DFSG.
- Software licensed under NonCommercial/NoDerivs cannot be named "Free Software"
- Authors who wish to use Attribution must use other licenses as BSD.
- Authors who wish to use Attribution-ShareAlike must use other licenses as GPL.

Some challenges: proliferation of licenses

- Vanity licenses: It has been a known problem in the community for a few years.
- A growing number of licenses increases exponentially the possible combinations and interactions.
- This fact makes it difficult to merge code from diverse sources, both for incompatibility issues and unacceptable clauses.
- It introduces juridical insecurity requiring lawyers, that it is what free licenses where trying to avoid in the first place (i.e. the EUPL license and EULAs).
- It favors FUD (Fear, Uncertainly, Doubt).

Some challenges: the "non-commercial" clause (1/3)

- It is frequently used, particularly in blogs and social sites, and it is now being adopted by some institutions.
- They are believed to protect the work against abusive use and opportunists (reselling or commercial exploitation by corporations, etc.).
- It is sometimes supported by alleging that protects investment (though some studies have challenged this view, since it restricts distribution).

Some challenges: the "non-commercial" clause (2/3)

It raises evil side effects:

- This clause does not distinguish between indirect commercial uses, self-funded projects, etc.
- It brings uncertainty about what is a commercial activity: when in doubt you might decide not to use it to avoid demands or consulting lawyers...
- Incompatibility with libre projects (i.e. Wikipedia).
- Causing confusion about the "free" concept: it promotes another kind of opportunism, by using viral marketing to create the impression that a work is free but not truly releasing it as a libre product.

Some challenges: the "non-commercial" clause (3/3)

- The author's right to decide the terms in which he shares his work is not at stake here.
- What it is rejected is the confusion and the subterfuge of presenting a work as a free product when it is not true.
- Use any license you like, but also use concepts with accuracy: do not label as *free/libre* or *copyleft* what it is not. Confusion damages free culture and benefit opportunists.

General Public License version 3

General Public License

- Based in Emacs license, first copyleft license (1986).
- GPL version 1 (1989). Generics, program-independent, "version 1 or later".
- GPL version 2 (1991). "Liberty or dead" clause (it prevents from patents threats).
- GPL version 3 (2007). Tivoization, patents and DRMs. Community discuss.

GPLv3 elaboration process

Public consultation process:

- It lasted eighteen months: from January 16, 2006 (first draft) to June 29, 2007 (final version).
- Four drafts.
- Five International Conferences (Boston, Porto Alegre, Barcelona, Tokyo and Brussels)

The most important change, compared to previous versions, was the re-elaboration of the license, since it was discussed and agreed by the community.

GPLv3 elaboration process

Public consultation process:

- It lasted eighteen months: from January 16, 2006 (first draft) to June 29, 2007 (final version).
- Four drafts.
- Five International Conferences (Boston, Porto Alegre, Barcelona, Tokyo and Brussels)

The most important change, compared to previous versions, was the re-elaboration of the license, since it was discussed and agreed by the community.

Changes in GPLv3

The newest GPL version does not invalidate previous versions or requires software to be licensed under the new version.

- Major changes
  - It DOES NOT prevent DRM implementations with GPL software, but it DOES allow interoperable software to be written with it.
  - More protection related to software patents
  - It neutralizes WIPO (*anti-circumvention*) laws which ban libre software (DMCA and EUCD).
  - It clarifies license compatibility (additional permissions)
- Minor changes
  - Adaptation to technological innovations.
  - Clarifications to make it easier to use and understand.
  - Better internationalization (*convey/distribution*)

Many changes, but fundamental principles remain.

Changes in GPLv3

The newest GPL version does not invalidate previous versions or requires software to be licensed under the new version.

- Major changes
  - It DOES NOT prevent DRM implementations with GPL software, but it DOES allow interoperable software to be written with it.
  - More protection related to software patents
  - It neutralizes WIPO (*anti-circumvention*) laws which ban libre software (DMCA and EUCD).
  - It clarifies license compatibility (additional permissions)
- Minor changes
  - Adaptation to technological innovations.
  - Clarifications to make it easier to use and understand.
  - Better internationalization (*convey/distribution*)

Many changes, but fundamental principles remain.

Digital Rights Management (DRM)

Also known as Digital Restrictions Mismanagement.
How GPLv3 works:

- Neutralize laws that prohibit (write or share) libre software (such as DMCA, EUCD)
- But not forbidding DRM with GPLed software.
- It's always possible to use GPLed code to write software that implements DRM
- But it's possible write interoperable software and bypass restrictions.
- Neutralize tivoization: require to provide with information or necessary data to install modified software on the embedded device.

GNU GPL does not restrict what people do in software; it just stops them from restricting others.

Software Patents

Protection against patent threats is implemented in GPLv2 through clause "Liberty or Death" (sec. 7).

- If GPLed code includes patents with incompatible restrictions, can't be distributed.
- Avoid "zombie" libre software (software would be free if patents won't exist anymore).

This clause remains in GPLv3.

Software Patents

GPLv3 adds stronger protection against patent threats through legal-engineering:

- Who distribute GPLed software must provide any patent rights to exercise the freedoms that the GPL grants him.
- If anyone intends to exercise a patent, your license is finished.
- Users and developers can work with GPLv3 software without worrying about anybody can sue for patent infringement.

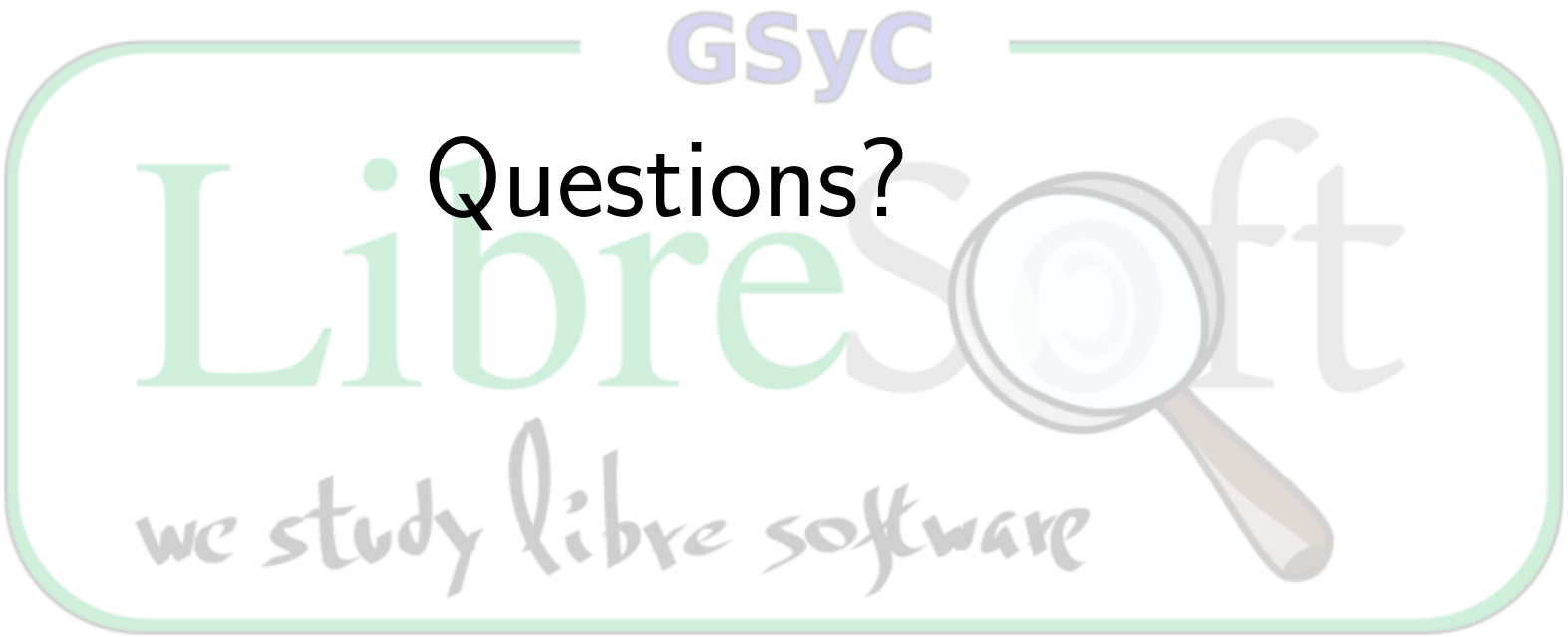
Compatibility

Compatibility == merge source code from different libre software licenses.

- GPLv3 increases compatibility with several free licenses (Apache, Affero).
- Allow additional requeriments:
  - Responsability: Allows add disclaimers or warranty notes.
  - Allows add restrictions about trademarks.

GPLv3 is more modular, more compatible, and will be compatible with different copyleft licenses.

Questions?



Questions?