Logiciel Libre
Cours 1 — Introduction

Stefano Zacchiroli
zack@pps.univ-paris-diderot.fr

Laboratoire PPS, Université Paris Diderot

2014–2015

Introduction to Libre Software

Master on Libre Software (URJC)

http://master.libresoft.es

Jesus M. Gonzalez-Barahona

jgb@gsyc.es
GSyC/LibreSoft, Universidad Rey Juan Carlos

September 2011

Introduction

There is a new guy in town

- GNU/Linux, Apache, GNOME, KDE, OpenOffice, etc. are very important, but...
- The really new thing is the libre software model:
    - Unprecedented combination of collaboration and competition.
    - Shift in emphasis from marketing to support and quality.
    - Classical assumptions about intellectual propriety are questioned.
    - End-users recover the control (instead of big software providers)
    - A new model for a new (global, networked) world?
- Last years have shown the feasibility of the model.

What is libre software?

In short free software guarantees:

- Freedom to use
- Freedom to study, and to adapt
- Freedom to redistribute
- Freedom to improve and release improvements

In other words, if you get it, you can...

- use it
- study and adapt it
- redistribute it
- improve it and release improvements

http://www.gnu.org/philosophy/free-sw.html

Free / libre / open source

- The definition is from Free Software Foundation, for free software ("Free Software Definition")
- But same applies to open source software ("Open Source Definition")
- To avoid discussion and missinterpretations, we will "libre software"
- Important consequence:
  To be able of modifying source code, it must be available.
- Lots of licenses: GPL, LGPL, BSD, Apache, MPL, etc.

`http://opensource.org/docs/osd`

Why this definition?

- Ethical concerns: the world should work this way
- Practical concerns: some actors benefit this way

Long discussions, that have reached some level of consensus:

- Free Software Definition (FSF)
- Debian Free Software Guidelines (Debian)
- Open Source Definition (OSI)

When libre software enters a new niche...

- It can become one of the first choices (GNU/Linux in operating systems, Apache in WWW servers, OpenOffice in office applications, etc.)
- It benefits from a lot of synergy (reuse of code, reuse of knowledge, reuse of distribution channels, etc.)
- Users gain competitive advantage:
  - Availability of source code makes improvements and customization possible in large scale (by in-house or subcontracted teams).
  - Standardization, but maintaining competition between providers.
  - No more per-use licenses.
  - Much more and better support (ensured by competition).
- Competition is the name of the game.

Consequences for the software industry

The software business is changing upside down (still slowly, but gaining momentum):

- Traditional software “manufacturers” will have to reinvent themselves completely (no more per-copy incomes).
- A whole new industry (based in support and libre development) will be needed as libre software gains market acceptance.
- It allows for (and encourages) competition in support, and even in the evolution of a piece of software.
- Users are benefited in several ways. Therefore, big pressure from end-users (including big companies) to switch to libre software.

Some specific impacts

- Cost: cost model radically different from proprietary software
- Openess: can be modified, can be inspected, can be studied
- Distribution: new distribution channels, new methods
- Development: “surprising” development models
- Maintenance and support: true competition

Mixture of two powerful mechanisms:

- Competition (using the same souce base)
- Cooperation (even non-voluntary)

Different actors, different visions

- End users (persons, companies, public administrations, etc.)
- Developer (or software producer)
- Software integration
- Service provider

Libre software for (large) end users

- Libre software is not necessarily better or worse. It is just different
- In several niches, we have already excelent products and companies supporting them.
- In many cases, the most cost-effective way of producing software.
- Special advantages when there is interest in long-term life cycles, vendor independence, multiplatform support, adaption to evolving technologies.
- If a powerful enough user (or group of users) needs to drive the technology, this is probably the best way to go.
- Many things can be done to promote a competitive libre software industry in a given niche. Many benefits are derived of such a promotion.

Conclusions?

- Still too few cases to be sure about future trends
- But there are interesting expectations
- The model seems to be economically and technically sound
- The model favors to the most competitive
- The model levels the field for smaller actors
- There is a lot of experimentation: new development, bussiness, user care, technology policy models
- Field with a lot of innovation: a good (and updated) knowledge about the environment is needed

Still too many issues to solve... or are they bussiness opportunities?

Why not learn about libre software?

Open source definition (annotated)

Foreword

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

[The text in all these slides was copied verbatim from the Open Source Definition (annotated version), as published by the Open Source Initiative]

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.

Rationale: By constraining the license to require free redistribution, we eliminate the temptation to throw away many long-term gains in order to make a few short-term sales dollars. If we didn't do this, there would be lots of pressure for cooperators to defect.

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.

Rationale: We require access to un-obfuscated source code because you can't evolve programs without modifying them. Since our purpose is to make evolution easy, we require that modification be made easy.

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.

Rationale: The mere ability to read source isn't enough to support independent peer review and rapid evolutionary selection. For rapid evolution to happen, people need to be able to experiment with and redistribute modifications.

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.

Rationale: Encouraging lots of improvement is a good thing, but users have a right to know who is responsible for the software they are using. Authors and maintainers have reciprocal right to know what they're being asked to support and protect their reputations. Accordingly, an open-source license must guarantee that source be readily available, but may require that it be distributed as pristine base sources plus patches. In this way, ünofficialçhanges can be made available but readily distinguished from the base source.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

Rationale: In order to get the maximum benefit from the process, the maximum diversity of persons and groups should be equally eligible to contribute to open sources. Therefore we forbid any open-source license from locking anybody out of the process. Some countries, including the United States, have export restrictions for certain types of software. An OSD-conformant license may warn licensees of applicable restrictions and remind them that they are obliged to obey the law; however, it may not incorporate such restrictions itself.

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.

Rationale: The major intention of this clause is to prohibit license traps that prevent open source from being used commercially. We want commercial users to join our community, not feel excluded from it.

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.

Rationale: This clause is intended to forbid closing up software by indirect means such as requiring a non-disclosure agreement.

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.

Rationale: This clause forecloses yet another class of license traps.

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.

Rationale: Distributors of open-source software have the right to make their own choices about their own software.
Yes, the GPL is conformant with this requirement. Software linked with GPLed libraries only inherits the GPL if it forms a single work, not any software with which they are merely distributed.

10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.

Rationale: This provision is aimed specifically at licenses which require an explicit gesture of assent in order to establish a contract between licensor and licensee. Provisions mandating so-called çlick-wrap" may conflict with important methods of software distribution such as FTP download, CD-ROM anthologies, and web mirroring; such provisions may also hinder code re-use. Conformant licenses must allow for the possibility that (a) redistribution of the software will take place over non-Web channels that do not support click-wrapping of the download, and that (b) the covered code (or re-used portions of covered code) may run in a non-GUI environment that cannot support popup dialogues.

Introduction to Libre Software

Master on Libre Software (URJC)
http://master.libresoft.es

Jesus M. Gonzalez-Barahona

{jgb,grex}@gsyc.es
GSyC/LibreSoft, Universidad Rey Juan Carlos

October 2012

Some consequences of the model

How libre software affects...

- End user (individual or company)
- Developer (or software producer)
- Integrator
- Maintenance and services

End user

End users can forget about...

- ...company monopolies
  (real competition, best products and services)
- ...producer ‘reliability’
  (future path ensured by product acceptance, source code availability, community dynamics)
- ...decision taking with few elements
  (software can be tested in real environments, with near-zero cost)
- ...dependence on provider’s strategies
  (many providers, community strategies, strategies follow clients)
- ...black boxes
  (no longer “blind confidence”)

End user (2)

What if users could...

- ...adapt/customize the product at will?
- ...have the latest release with (very) low cost?
- ...fix all the problems (or hire someone to fix them)?
- ...decide on the future evolution of the product?
- ...contract the (complete) integration of the best products in a given area?
- ...buy complete auditing for each product by independent third parties?

End user (3)

A large portion of the control moves to the user
(from the producer of the software)

Developer (or software producer)

Libre software changes the rules of the game:

- Opportunities for competing while being small
- Easier (and cheaper) to acquire front-wave technology
- Can take advantage of the work of your competitors (but they can do the same!)
- External contributors can be found (in many cases, at a fraction of the usual cost, because of win-win relationships)
- Distribution channels are cheaper, and truly global
- Feasible to become reference application in a niche

Developer (or software producer) (2)

Where does the money come from? (sustainability)

- The producer enjoys the best knowledge about its product
- Producer can be the “most visible point”, if image is cared of
- Custom-made development, modifications, customizations
- “In depth” support (bug fixing, preference in access to new releases, new features)

Assuming there is a need for a software product
and there is money ready for supporting that need
some developer/producer will benefit from the situation
(if both parts are put in contact)

Integrator

Maybe the best placed actor:

- All libre software products available (without the constraints of proprietary licences!)
- If products “don’t fit” you can adapt them (source code is available, interoperability is always possible)
- Pieces of products, or full products, or anything in the middle, can be integrated
- No more black boxes: everything is transparent

They can build on top of the work of others, with similar constraints and possibilities to those others

Services and maintenance

- Similar conditions than the producer
- Competition in the maintenance business
- Added-value of services is better appreciated (the base cost of the program is low)
- Good knowledge of the state of the art is important (good idea to have good links with libre software projects)
- New business models: advising on releases and combination of programs, information about new development, project management, etc.
- The most diverse and massive kind of business right now

Some conclusions

- Libre software changes the rules of the game
- It is important to understand (and get advantage) of those rules
- Still learning effects and mechanisms
- Many opportunities to discover new effects, and take advantage of them

To probe further

- “SME Guide to Free Software”, by Carlo Daffara
  `http://guide.flossmetrics.org`

Myths

“Copyleft is against intellectual property”

"Free software has no owners"

"Free software developers have the obligation to publish all their modifications"

"Free software can be used for genocide"

"Free software is communist"

“Free software is libertarian”

"Free software starts with Richard Stallman"

“Free Software is more secure”

Myths (cont.)

“Free Software is [necessarily] better”

(from a feature/software quality point of view)