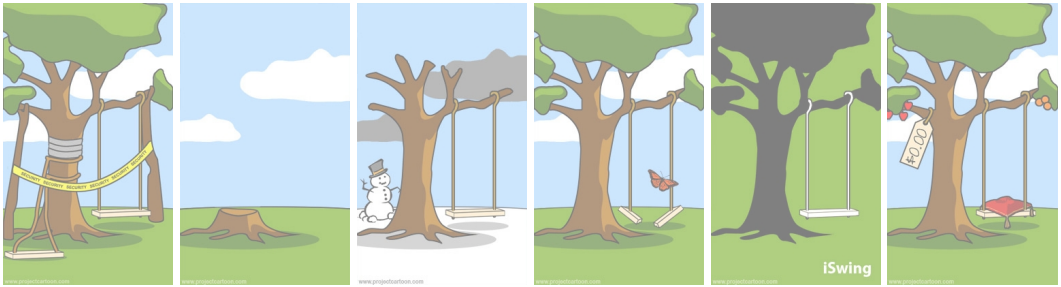
Software Engineering

16. Open-Source Software | Thomas Thüm | June 21, 2022

Software Engineering
Programming Languages

ulm university universität
uulm

Open-Source Software



how patches were applied	how it was supported	when it was delivered	how it performed under load	what marketing advertised	the open source version
software evolution	**software maintenance**	**configuration management**	**compilation and static analyses**	**software product lines**	**open-source software**

Lecture Overview

1. Open-Source Software
2. Open-Source Principles
3. Open-Source Licenses

Lecture Contents

Open-Source Development [Sommerville]

Open-Source Development

"**Open-source development** is an approach to software development in which the source code of a software system is published and volunteers are invited to participate in the development process."

Open-Source Development

- assumption in the early days: code developed by small core group
- today: Internet to recruit volunteer developers (often users)
- volunteers contribute with bug reports, feature requests, changes to the software (cf. pull requests)
- core group controls changes to main repo
- opposite: proprietary software / closed-source development

Internet iff Open Source

- Internet used to distribute source code and recruit developers
- Internet builds on open-source software

Examples

- operating system Linux (for most servers)
- operating system Android (for most clients)
- web server Apache
- database management system mySQL
- browser Firefox
- image editor GIMP
- media player VLC
- screen recording software OBS
- video editing software Shotcut
- runtime env. and standard library OpenJDK
- development environment Eclipse

Central Questions for Each Software Project [Sommerville]

First Question

"Should the product that is being developed make use of open-source components?"

Example Criteria

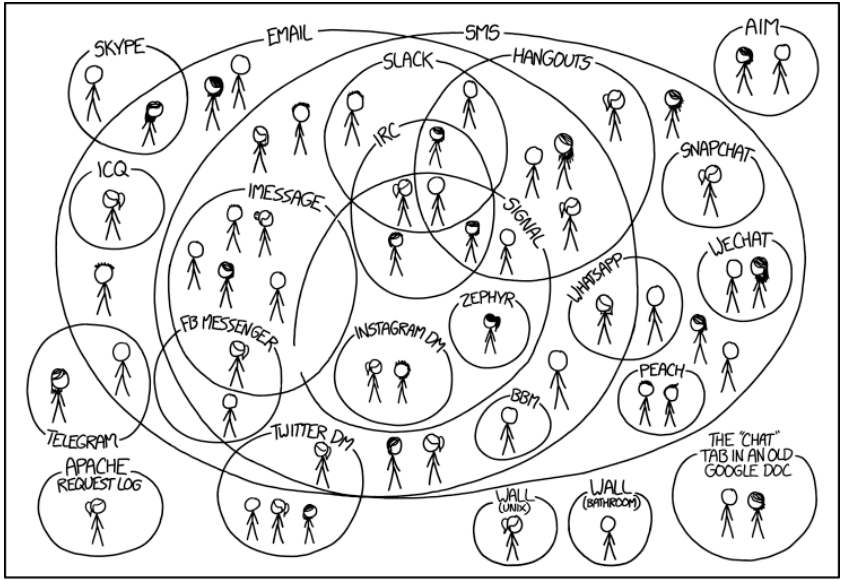
- are related open-source components available?
- is their quality sufficient?
- will they be maintained?
- what is the effort to integrate them?
- is it cheaper to modify or redevelop them?
- how are they licensed?

Second Question

"Should an open-source approach be used for its own software development?"

Example Criteria

- business model: how to earn money then?
- support? consulting? proprietary extensions? – *rich relatives or friends? unconditional basic income (bedingungsloses Grundeinkommen)?*
- what additional value do you have from opening the source?
- more customers, recognition, collaborations?
- does it reveal confidential business knowledge?



I HAVE A HARD TIME KEEPING TRACK OF WHICH CONTACTS USE WHICH CHAT SYSTEMS.

Richard Stallman's Four Freedoms [Stallman's 2014 TEDx Talk]

Richard Matthew Stallman aka. RMS (born 1953)

Four Freedoms by Richard Stallman

- 0. freedom to run it for whatever purpose
- 1. freedom to study and modify the program (source code is available)
- 2. freedom to redistribute it (needed for non-programmers)
- 3. freedom to redistribute or even sell modified versions
- all four needed that the user controls the program
- otherwise the program controls the users

Free Software [Stallman's 2014 TEDx Talk]

Free Software

- free stands for freedom (often confused with "free of charge")
- free software: demand for freedom
- open-source software: better code quality (promoted by Eric S. Raymond)
- today largely replaced with term open-source software
- more inclusive term?
  free and open-source software (FOSS)

Free Software vs Freeware

- free software: free as in free speach
- freeware: free as in free beer

free software can be adware or shareware

Free Software Foundation [fsf.org]

- founded by Richard Stallman in 1985
- first ten years: employ developers for the operating system GNU
  Linux = GNU + Linux kernel
- later: promoting free software, working on legal issues

Buying Free Software

Open-Source Software

Lessons Learned

- Open-source development and open-source software
- Examples and central questions around open source
- Free software foundation: four freedoms, RMS, FOSS
- Further Reading: Sommerville, Chapter 7.4 Open-Source Development
- Further Material: Stallman's 2014 TEDx Talk, Stallman's lecture in Ulm

Practice

- Take any of the messengers on the prior slide and report whether it is open source.
- How many messengers do you use?
- Share both with your colleagues in Moodle

Lecture Contents

1. Open-Source Software

2. Open-Source Principles
   - Rights and Duties
   - Collective Ownership
   - Recommendations for Companies
   - Copyleft and Copyright
   - The Impact of Licensing Issues
   - Lessons Learned

3. Open-Source Licenses

Rights and Duties

Who owns the code? [Sommerville]

"Although a fundamental principle of open-source development is that source code should be freely available, this does not mean that anyone can do as they wish with that code. Legally, the developer of the code (either a company or an individual) owns the code. They can place restrictions on how it is used by including legally binding conditions in an **open-source software license**."

What if there is no license?

- you are allowed to read the code
- you are **not allowed** to use, modify, distribute it
- you need to contact the owners to negotiate

Understanding Software Licenses [fossa.com]

"Anyone who works with open-source software (OSS), whether as a developer, a contributor, or a business, has to know at least a little bit about open source licenses. In a nutshell, an open source license tells you what you can and can't do with the open source code. And if using the code comes with any requirements and/or responsibilities, the license outlines those as well."

What's Next?

- principles of software licenses
- examples of famous software licenses

Collective Ownership

Sercan Leylek (2016) [wordpress.com]

"Every programmer is an author."

Mosher's Law of Software Engineering [twitter.com]

"Don't worry if it doesn't work right. If everything did, you'd be out of a job."


PROBLEM: ONE OF THE VOLUNTEER DEVELOPERS HAS A DATE THIS WEEKEND. DATES LEAD TO ROMANCE. ROMANCE LEADS TO ORPHANED PROJECTS.
DEBIAN LINUX HQ
WHAT'S THE PLAN?
WE'RE HIRING HIM A RELATIONSHIP COACH. HE'S LIKE WILL SMITH IN "HITCH," BUT HE ONLY GIVES BAD ADVICE.
DEBIAN LINUX HQ
OKAY, REMEMBER: THE KEY TO CONVERSATION IS CONSTRUCTIVE CRITICISM.
YOU NEED TO SHOW YOU'RE SMART ENOUGH TO SOLVE HER PROBLEMS.
MAKES SENSE.

Recommendations for Companies

Motivation

- basically every company needs software
- even non-software companies need software
- more and more companies even transition into software companies (e.g., car manufacturers)
- large amounts of money spend on software licenses (e.g., Microsoft Windows and Office, Adobe Acrobat)
- even if open-source software is for free, resources are necessary to prevent license violations within a company
- and who does the maintenance? who pays for it?

Bayersdorfer 2007: [Sommerville]

- track information about downloaded and used open-source components (e.g., storing licenses)
- understand how a component is licensed before it is used
- study the open-source project to predict its future evolution
- educate developers about open source and open-source licensing 
- auditing of open-source software to detect violations
- if you rely on open-source products, support their development

Copyleft and Copyright

Copyleft

- right to freely distribute and modify intellectual property
- obligation that the same rights (license) apply to derivative work
  - **weak copyleft**: only applies to formerly-licensed parts (e.g., a library)
  - **strong copyleft**: applies to the complete software (deprecatory: viral effect)
- license without copyleft is called **permissive**
- implemented with copyright laws

Copyright (Urheberrecht)

- kind of intellectual property, such as patents and trademarks (gestiges Eigentum, Patent, Marke)
- owner of creative work has exclusive right over copies for a limited time
- includes distribution, reproduction, public performance, derivative work
- copyright is often granted by national laws

Examples for Copyleft Licenses

- GNU General Public License (GPL) for software
- Creative Commons share-alike license for documents and pictures

Decision for copyleft is independent of:

- decision to allow/forbid commercial use
- decision about the fee to get the source code

The Impact of Licensing Issues [heise.de]

Ruby on Rails: Durch Lizenzproblem entfallene Library erzeugt Dominoeffekt

Eine halbe Million Open-Source-Projekte sind wohl von dem Chaos betroffen, das durch eine erst falsch lizenzierte, dann zurückgezogene Library entstanden ist.

Lesezeit: 5 Min. In Pocket speichern 376

March 2021: The Story of mimemagic

- Ruby library mimemagic is part of Ruby on Rails and was licensed under MIT license
- 580,000 repositories on Github use Ruby on Rails under the same license
- mimemagic is using another library under GPL license
- GPL has a copyleft: mimemagic must be published under GPL too
- old versions of mimemagic have been removed
- new versions use GPL license
- 580,000 projects would need to change from MIT to GPL and upgrade to the new version

Open-Source Principles

Lessons Learned

- Need for and impact of software licenses
- Ownership and code without any license
- Recommendations for companies
- Copyleft and its impact on software
- Further Reading: Sommerville, Chapter 7.4 Open-Source Development

Practice

- Hypothesis: open-source software is for free
- *Fight for Your Right*
- Collect arguments for or against this claim on Moodle

Lecture Contents

1. Open-Source Software

2. Open-Source Principles

3. Open-Source Licenses
   - MIT License
   - GPL: GNU General Public License
   - LGPL: GNU Lesser General Public License
   - Compatibility of Software Licenses
   - Re-Licensing and Dual Licensing
   - Lessons Learned

MIT License [fossa.com]

Profile

- released: about 1987
- publisher: Massachusetts Institute of Technology
- classification: permissive
- most popular license
- prominent use: Ruby on Rails (server-side web application framework), Node.js (JavaScript runtime environment)

Selected Terms and Conditions

- permits reuse within proprietary software
- license shipped with distribution or relicensing (even to proprietary licenses)
- "provided as-is, without warranty"

The Full License Text

"Copyright (c) <year> <copyright holders>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE."

GPL: GNU General Public License [gnu.org]

Profile

- releases: 1989 (GPLv1), 1991 (GPLv2), 2007 (GPLv3)
- author/publisher: Richard Stallman, Free Software Foundation
- classification: strong copyleft
- prominent use: Linux kernel (GPL-2.0-only), GNU Compiler Collection (GCC)

Selected Terms and Conditions

- software using (modified) GPL software must be released under GPL (copyleft)
- for any sale or distribution: binaries must be accompanied by source code
- for private/interal use: no obligations
- commercial redistribution allowed
- GPL software (Linux/GCC) may be used for commercial purposes

Three Main Versions

- GPL-1.0-only and GPL-1.0-or-later
  - publishing binary only was allowed
  - applies to whole distributable
- GPL-2.0-only and GPL-2.0-or-later
  - fixed above problems
  - relaxed version LGPL, motivated by C standard library (libc)
- GPL-3.0-only and GPL-3.0-or-later
  - improved compatibility with other licenses (e.g., Apache)

LGPL: GNU Lesser General Public License [gnu.org]

Profile

- releases: 1991 (LGPLv2), 1999 (LGPLv2.1), 2007 (LGPLv3)
- formerly known as: GNU Library General Public License
- publisher: Free Software Foundation
- classification: weak copyleft

Selected Terms and Conditions

- similar to GPL, but less restrictive
- software using (modified) LGPL software components **does not have to** be released under LGPL
- modifications of LGPL software components **must** be released under LGPL
- freedom only for LGPL components

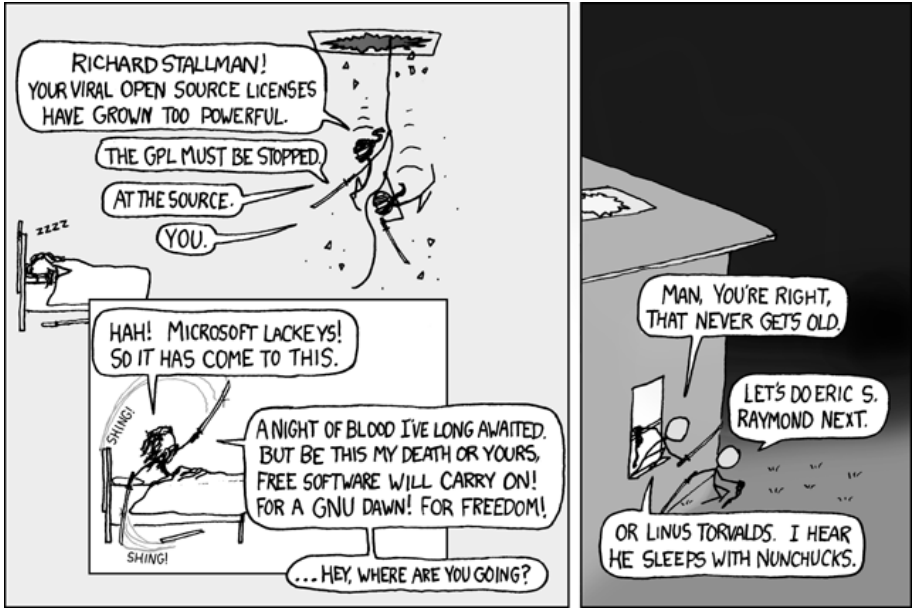
Three Main Versions

- LGPL-2.0-only and LGPL-2.0-or-later
- LGPL-2.1-only and LGPL-2.1-or-later
- LGPL-3.0-only and LGPL-3.0-or-later

What's the main problem of LGPL?

It contains "GPL"!

LGPL often confused with GPL in industry.


RICHARD STALLMAN! YOUR VIRAL OPEN SOURCE LICENSES HAVE GROWN TOO POWERFUL.
THE GPL MUST BE STOPPED.
AT THE SOURCE.
YOU.
ZZZZ
HAH! MICROSOFT LACKEYS! SO IT HAS COME TO THIS.
SHING!
A NIGHT OF BLOOD I'VE LONG AWAITED. BUT BE THIS MY DEATH OR YOURS, FREE SOFTWARE WILL CARRY ON! FOR A GNU DAWN! FOR FREEDOM!
SHING!
...HEY, WHERE ARE YOU GOING?
MAN, YOU'RE RIGHT, THAT NEVER GETS OLD.
LET'S DO ERIC S. RAYMOND NEXT.
OR LINUS TORVALDS. I HEAR HE SLEEPS WITH NUNCHUCKS.

Compatibility of Software Licenses

Problem

- most licenses have similar goals
- still: sometimes legally impossible to combine software with different licenses
- incompatibilities often arise from copyleft clauses

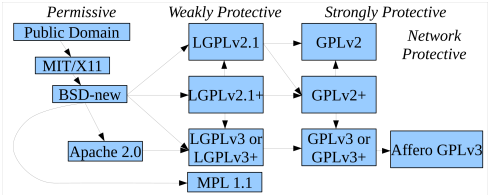
Desired Incompatibilities

- proprietary licenses are typically specific to a program and incompatible to each other
- copyleft licenses are incompatible to proprietary and permissive licenses
- software licensed with **GPL** cannot be used within **MIT**-licensed software

Undesired Incompatibilities

- software licensed with **GPL-2.0-only** cannot be used within **GPLv3**- or **LGPLv3**-licensed software (cf. Linux kernel) $\Rightarrow$ most projects use **GPL-2.0-or-later** instead
- software licensed with **GPLv3** cannot be used in material licsensed as creative commons **BY-SA 4.0** [creativecommons.org]

Overview on Compatibilities



Re-Licensing and Dual Licensing

How to solve a license incompatibility?

- remove the software everywhere
- re-implement (parts thereof)
- give it a new license (as replacement or in addition to the old license)

Re-Licensing

- changing the license to another license
- requires consent by all developers
- in practice 100% often not feasible: 80–95%

Re-Licensing of the VLC Project

- removed from the Apple App Store
- re-licensed from GPLv2 to LGPLv2
- later re-licensed to Mozilla Public License

Dual Licensing

- owner is not bound to the own copyleft/license
- a software can have multiple licenses
- same rules as for re-licensing
- term also used for 3 and more licenses

Mozilla's Firefox Browser

- Netscape's Communicator 4.0 released under the Mozilla Public License
- incompatibility to GPL
- dual licensing as MPLv1.1/GPLv2/LGPLv2.1

Open-Source Licenses

Lessons Learned

- Software licenses: MIT, GPL, LGPL
- License incompatibilities: desired and undesired
- Re-licensing and dual licensing

Practice

- Choose a software license (e.g., CC0, CC-BY, CC-BY-SA, CC-BY-NC, CC-BY-NC-SA, CC-BY-ND, CC-BY-NC-ND, Unlicense, BSD, Apache, MPL, EPL, Beerware) and create a profile similar to that for MIT/GPL/LGPL
- It is absolutely fine to only partially fill the profile
- Start a profile or continue an existing one in Moodle