Logiciel Libre
Cours 4 — Licenses

Stefano Zacchiroli
zack@irif.fr

Laboratoire IRIF, Université de Paris

2019–2020

URL https://upsilon.cc/zack/teaching/1920/loglib/

Outline

Outline

Why Do I Need a License? (redux)

User point of view

- Copyright covers software.
- Copyright is “closed by default”.
- As user: if you don’t have a license for some code, you can’t use it (legally).

No License Required?

Author point of view

- Copyright kicks in as soon as someone creates a “tangible” (expressible) work.
- In absence of any licensing declarations, don’t allow any use (“all rights reserved”).
- Without a license your (potential) users can’t use your software.
- You need to offer at least *some rights*.

FOSS licenses are legal hacks

- FOSS licenses are legal hacks: they behave as other copyright licenses, but instead of restricting user rights, they grant more (and very specific) rights
- in particular: FOSS licenses grant enough rights to ensure users enjoy the 4 freedoms (run, study, copy, modify)
- that does *not* mean "do anything you want"; FOSS licenses can (and do) impose specific conditions
  - if you do not respect them, the license does not apply to you and you fallback to copyright default: "all rights reserved"

Note: FOSS is hence not against "IP". In fact, FOSS licenses *use* copyright law to guarantee software freedom.

Licenses are constitutions for FOSS communities

- Software licenses are social contracts just as much as they are legal documents
- When you choose a license, you are charting a course for the future
- You are often establishing a relationship to a larger community
- Not purely about mechanical and legal choices
- It is very difficult to change later: it is worth to invest time into understand licensing before choosing

FOSS License Example

Implementing a basic Free Software license might be very easy: [1]

Example

Copyright © 2019 Foobar Developers.

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

That's it!

1. don't use *this* as an actual FOSS license, better examples follow

FOSS licenses implement software freedom

FOSS licenses are the legal mechanism used to implement the 4 freedoms for software users

When you receive a Free Software you get:

- Freedom #0, to run the program, for any purpose
- Freedom #1, to study how the program works, and change it
- Freedom #2, to redistribute copies
- Freedom #3, to improve the program, and release improvements

Note: *all* four freedoms must be granted *at the same time* for the software to be considered FOSS.

Recurring concepts in FOSS licensing

- 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 on other programs, reusing parts of it (in binary or source code form)
- Authorship attribution: The obligation of recognizing the authorship of a work when using it or applying any change, such as deriving or redistributing it.
- Licensing: the act of choosing a (FOSS) license for a specific copy of a software
  - it is a privilege of the copyright holder(s)
  - note: different copies of the *same* software might be distributed under different licenses

Software remains "owned" (in the "IP" sense) by the copyright holder. Users only get specific rights, determined by the license.

Restrictions and FOSS

Are there permissible restrictions in FOSS licenses?

Restrictions and FOSS

Are there permissible restrictions in FOSS licenses?

Yes: everything that does not get in the way of the 4 freedoms and/or the Open Source Definition is acceptable.
In practice, deciding what is OK and what is not is not always clear cut, is often not codified in guidelines, and the decisions may vary across gatekeepers (FSF/OSI/Debian/etc).

Commonly accepted restrictions are:

- mandatory attribution of authors (as long as attribution does not impede normal use of the work)
- transmission of freedoms (see copyleft later)
- protection of specific freedoms (e.g., access to source code or prohibition of “technical measures”, DRM)

FOSS license categories

FOSS licenses can be classified according to the conditions they impose in exchange of software freedom.
We identify the following macro-classes of FOSS licenses:

- Lax permissive (AKA “permissive”)
- Scope-limited reciprocal (AKA “weak copyleft”)
- Reciprocal (AKA “strong copyleft”)

Note: “more strict” licenses are not “less free” than others. Even the most strict FOSS licenses are incomparably more permissive than proprietary software licenses

Exercise

Compare any FOSS license to the EULA (End User License Agreement) of Microsoft Windows.

Academic licenses

- Historically relevant subset of lax permissive licenses
- The simplest licenses: very few restrictions
- Mandating only attribution (keep names and copyright notice)
- Available for all uses, including use in proprietary software
- Originally written for and popularized by universities

Examples: MIT, BSD, ISC

Lax permissive licenses

- Include explicit grant of patent license (in modern variants)
- Available for almost all uses, including use in proprietary products

Examples: Apache (the license)

Reciprocal licenses

- Require that derivative works maintain the same license
- Usually require binary distributions to be accompanied by complete and corresponding source code (CCS)
- Also known as “strong copyleft” or just “copyleft”
- Historically called “viral licenses”, as a denigration tactic
  - If reciprocally licensed code is incorporated, then the application is “infected” and must be released as a whole under the same license

Examples:

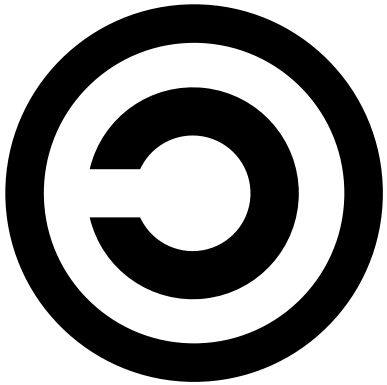
- GPL, AGPL
- CC BY-SA (for non-software works)

Scope-limited reciprocal licenses

- Like reciprocal licenses, but with limitations on the scope of which parts of a derived work fall under the license terms
  - changes to the "main work" falls under the license terms
  - "additional works" that happen to be used with/added to/embedded with the main work do not
- They vary in the way the boundaries between main and additional works is defined
- According to the denigratory analogy: "virality" is limited to the main work
- Also known as: "weak copyleft"

Examples: MPL, CDDL, LGPL

What is copyleft?



Copyleft is a strategy of utilizing copyright law to pursue the policy goal of fostering and encouraging the equal and inalienable right to copy, share, modify and improve creative works of authorship.

Copyleft (as a general term) describes any method that utilizes the copyright system to achieve the aforementioned goal. Copyleft as a concept is usually implemented in the details of a specific copyright license, such as the GNU General Public License (GPL) and the Creative Commons Attribution Share Alike License.

Copyright holders of creative work can unilaterally implement these licenses for their own works to build communities that collaboratively share and improve those copylefted creative works.

— *http://copyleft.org/*

What is copyleft? (cont.)

- Granting the four freedoms is enough to guarantee users will get them only for a specific copy of the work
  - how about further downstream redistribution?
  - how about derived works?
  - (how about future versions?)
- Copyleft makes sure that all users receiving a copy of the program, no matter how modified, also enjoy the four freedoms
- The copyleft clause might have diverse implementations but all of them share the same concept: distribution of any version of this program must preserve user freedoms.
- Copyleft is also an industrial strategy: it ensures a level-playing field (contrary to lax permissive licenses), that promotes co-opetition
- On the other hand copyleft does preclude *some* business models, and for that reason it might get backlash

License compatibility

- Two licenses are compatible if a joint derivative work (i.e., a work containing code released under each license) could be legally distributed
  - ideally as FOSS, although the notion of compatibility is general
- Compatibility is determined by comparing restrictions imposed by all involved licenses
  - e.g., GPL and MPL version 1.1 are incompatible (i.e., it is impossible to integrate code released under the two licenses without violating the terms of at least one of them)
    - GPL: user must convey the entire program's source code "under the terms of *this* (= GPL) License"
    - MPL 1.1: modifications are "governed by the terms of *this* (= MPL) License"
- A dependent variable, that does not affect compatibility *per se*, is the resulting license under which the joint derivative work will be redistributed
  - e.g., GPL and BSD licenses are compatible, but the resulting joint work will be under the terms of GPL only

Dual- (or multi-) licensing

Distribute software under two (or more) different sets of licenses.

The expression is unfortunately overloaded to express different notions:

- license segregation: different licenses apply to different copies of the same program (e.g., for proprietary relicensing business models)
- user choice: different, alternative (OR-ed) licenses apply to the same copy of the software; the user choose the license
  - degenerate case: "version N or above" clauses. The user can choose *which version* of the license apply to them

Motivations:

- License compatibility (e.g., Perl, Firefox)
- Business models based on market segregation (e.g., MySQL, OCaml)
- Future-proof license-based strategies

Should I write my own license?

Should I write my own license?

NO.

Bad *ad-hoc* licensing example: ipfilter (2000)

```
/*
 * Copyright (C) 1993-2000 by Darren Reed.
 *
 * The author accepts no responsibility for the use of this software
 * and provides it on an ``as is'' basis without express or implied
 * warranty.
 *
 * Redistribution and use in source and binary forms are permitted
 * provided that this notice is preserved and due credit is given
 * to the original author and the contributors.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
 *
 * I hate legalese, don't you?
 */
```

ipfilter license “clarification” (2001)

```
/*
 * Copyright (C) 1993-2000 by Darren Reed.
 *
 * The author accepts no responsibility for the use of this software
 * and provides it on an ``as is'' basis without express or implied
 * warranty.
 *
 * Redistribution and use in source and binary forms are permitted
 * provided that this notice is preserved and due credit is given
 * to the original author and the contributors.
 *
 * Yes, this means that derivative or modified works are not
 * permitted without the author's prior consent.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
 *
 * I hate legalese, don't you ?
 */
```

Theo de Raadt announces ipfilter replacement

```
Date: Tue, 29 May 2001 19:13:11 -0600
From: Theo de Raadt <deraadt@cvs.openbsd.org>
Subject: ipf

sometime in the next 20 hours, i will be removing ipf from the
source tree since it does not meet our freedom requirements,
as have been outlined in policy.html and goals.html since the
start of our project.

we will have to work on an alternative.
```

https://groups.google.com/d/msg/fa.openbsd.tech/q3b--naHTF0/iERRvuKkTFEJ

ipfilter case

- The real problem is that code with a non-free license was incorporated into the core of a free operating system
- Carelessness with licenses invites trouble

Why you should not write your own license

Many people have attempted to write their own FOSS licenses, especially in the early days, but:

- You will probably get it wrong (e.g., Artistic License 1.0)
- Your license will not immediately be approved or recognized by FOSS license gatekeepers—OSI, FSF, Debian—limiting adoption (of both the license and *your code*)
- You will contribute to license proliferation

License proliferation

- Vanity licenses: known problem in the community in the early years
- A growing number of licenses increases quadratically the possible combinations and interactions
- That, in turn, makes difficult to merge code from diverse sources, both for incompatibility issues and unacceptable clauses
- It introduces juridical uncertainty requiring lawyers, which is what "public" licenses were trying to avoid in the first place
- It favors FUD (Fear, Uncertainly, Doubt) about FOSS complexity

See: Open Source Initiative, *"The License Proliferation Report"*, 2006, `http://opensource.org/proliferation`

Which license should I choose then?

Two main situations: contributing to an existing FOSS project v. creating a new one from scratch

a. If you contribute to an existing FOSS project: just use the current license of the project for your contributions (often you don't get to choose anyhow).

b. If you create a new FOSS project: choose a license that is
   1. approved by *both* OSI [2] and FSF [3], and
   2. popular, and
   3. matches the target community/strategy of the project

2. OSI license list: `https://opensource.org/licenses`
3. FSF license list: `https://www.gnu.org/licenses/license-list.en.html`

Outline

Popular and noteworthy licenses

- Lax permissive (AKA “permissive”)
  - BSD 3-Clause “New” or “Revised” license
  - BSD 2-Clause “Simplified” or “FreeBSD” license
  - Apache License 2.0
  - MIT license
  - ISC License
- Scope-limited reciprocal (AKA “weak copyleft”)
  - GNU Lesser General Public License (LGPL), versions 2.1 and 3
  - Mozilla Public License (MPL), version 2.0
- Reciprocal (AKA “strong copyleft”)
  - GNU General Public License (GPL), versions 2 and 3
  - GNU Affero General Public License (AGPL), version 3

Outline

BSD License — origins

- BSD (Berkeley Software Distribution) is a Unix flavor developed by University of Berkeley (CA).
- BSD Unix was released under the terms of a “minimalistic” license, which permits both source and binary redistribution, with or without modifications, without any other restriction.
- Historical origin of the most liberal tradition in FOSS, opposing the use of copyleft as a strategy to liberate more software
  - intuition: favor “developers’ freedoms” over “users’ freedoms”
- Several revisions of the license exist
- Each revision is in fact a template, where copyright notices should be properly instantiated

Modern BSD Licenses

- Descendants of the original BSD license
- Very popular (BSD userland, PF, TCP/IP, OpenSSH, TCL/Tk...)
- You may redistribute the work, in any form (source or binary), as long as you preserve copyright notices
- Includes "as is" and "no warranty" clauses
- "Liberal (= libertarian) license": no control over software evolution

BSD License — advantages

- BSD places minimal restrictions on developers and future software evolutions
- This allows BSD code to remain FOSS or become integrated into proprietary solutions
- Little legal complexity (unlike the *GPL family of licenses)
- It allows developers and companies to spend their time creating and promoting good code rather than worrying about license violations

Prior BSD License (1988)

Copyright (c) <year> <copyright holder>.
All rights reserved.

Redistribution and use in source and binary forms are permitted provided that the above copyright notice and this paragraph are duplicated in all such forms and that any documentation, advertising materials, and other materials related to such distribution and use acknowledge that the software was developed by the <organization>. The name of the <organization> may not be used to endorse or promote products derived from this software without specific prior written permission.
THIS SOFTWARE IS PROVIDED ''AS IS'' AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

- *de facto* obsolete
- approved by: FSF, Debian
- not approved by: OSI (because better recent alternatives exist)
- GPL compatible

Digression — warranty and disclaimer

- Software by itself is not a consumer product
- When software is (combined into) a consumer product, disclaimers are ineffective
- “As Is”: your are accepting item in the actual state with all its faults

Example — BSD warranty disclaimer

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 OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER 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.

(yes: it's all ALL CAPS)

4-clause / Original BSD license (1990)

Copyright (c) <year>, <copyright holder>
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. *All advertising materials mentioning features or use of this software must display the following acknowledgement:*
   This product includes software developed by the <organization>.
4. *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.*

[as-is + no warranty disclaimer]

4-clause / Original BSD license (cont.)

- infamous advertisement clause (AKA: “badgeware”)
  - advertisement notices escalation, up to 70 in NetBSD
  - further restriction
- *de facto* obsolete
- approved by: FSF, Debian
- not approved by: OSI
- GPL incompatible

3-clause / Revised / New BSD License (1999)

Copyright (c) <year>, <copyright holder>
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.*

[as-is + no warranty disclaimer]

3-clause / Revised / New BSD License (cont.)

- intuition: 4-clause without advertisement clause
- popular permissive license
- approved by: FSF, OSI, Debian
- GPL compatible

2-clause / FreeBSD / Simplified BSD License

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:

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.*

[as-is + no warranty disclaimer]

The views and conclusions contained in the software and documentation are those of the authors and should not be interpreted as representing official policies, either expressed or implied, of the FreeBSD Project.

2-clause / FreeBSD / Simplified BSD License (cont.)

- intuition: 3-clause without the non-endorsement clause
- used by the FreeBSD project
- approved by: FSF, OSI, Debian
- GPL compatible

Popular BSD-like licenses

- ISC
- MIT

ISC: the shortest FOSS license

Copyright (c) 4-digit year, Company or Person's Name

Permission to use, copy, modify, and/or distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED "AS IS" AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY SPECIAL, DIRECT, INDIRECT, OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

- Functionally similar to the 2-clause BSD
- Language made "unnecessary" by Berne Convention removed.[4]
- BIND, DHCP, and preferred license by the OpenBSD project
- approved by: FSF, OSI, Debian
- GPL compatible

4. according to `http://www.openbsd.org/policy.html`

The MIT License

The MIT License (MIT)
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.
[as-is + no warranty disclaimer]

- Functionally similar to the 2-clause BSD.
- It doesn't contain an explicit notice prohibiting the use of the name of the copyright holder in promotion
- It states more explicitly the rights given to the end-user
- Very popular: originally X.org, now web frameworks (e.g., Node), 45% of GitHub projects (2015)
- approved by: FSF, OSI, Debian
- GPL compatible

Other BSD-like licenses

- Zope Public License 2.0
  - Used by the Zope application server and related products
  - Similar to the BSD license, but with explicit clauses that prohibits the use of Zope Corporation trademarks and require documentation of changes
  - approved by: FSF, OSI, Debian
  - GPL compatible

    `http://opensource.org/licenses/ZPL-2.0`

Apache License

- Old versions: 1.0 (original) and 1.1 (ASF, 2000)
  - `https://www.apache.org/licenses/LICENSE-1.0`
- An extension of the 3-clause BSD license
- Permits integration into closed source projects

- Apache License 2.0 (January 2004): permissive license.
  - Make the license easier for non-ASF projects to use
  - Explicitly grants patent rights where necessary to operate, modify and distribute the software (§3)
  - Patent retaliation (terminating the license upon the initiation of a lawsuit)

`https://www.apache.org/licenses/LICENSE-2.0`

Apache License 2.0

- popular license
  - ≈150 projects hosted by the Apache Software Foundation (2015)
  - over 8'000 non-ASF projects located at SourceForge are available under Apache License (2012)
  - 25% of Google Code projects, including Android user space (2008); 11% of GitHub (2015)
- approved by: FSF, OSI, Debian
- compatible with GPLv3
- incompatible with GPLv2

Apache 2.0: patent license

§3. Grant of Patent License

Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable, patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work , where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) [...].

Apache 2.0: patent retaliation

§3. Grant of Patent License

[. . .] If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

Outline

Extreme liberal licensing

How far can we go with liberal licensing?

i.e., maximizing user freedoms and minimizing constraints

An example: WTFPL License

DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
Version 2, December 2004

Copyright (C) 2004 Sam Hocevar <sam@hocevar.net>

Everyone is permitted to copy and distribute verbatim or modified copies of this license document, and changing it is allowed as long as the name is changed.

DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. You just DO WHAT THE FUCK YOU WANT TO.

- Irreverent text, on purpose
- Licensees are encouraged to do what the @*&%!#* they want to
- Not very popular, not a good choice for software
- Approved by: FSF, Debian
- Not approved by: OSI

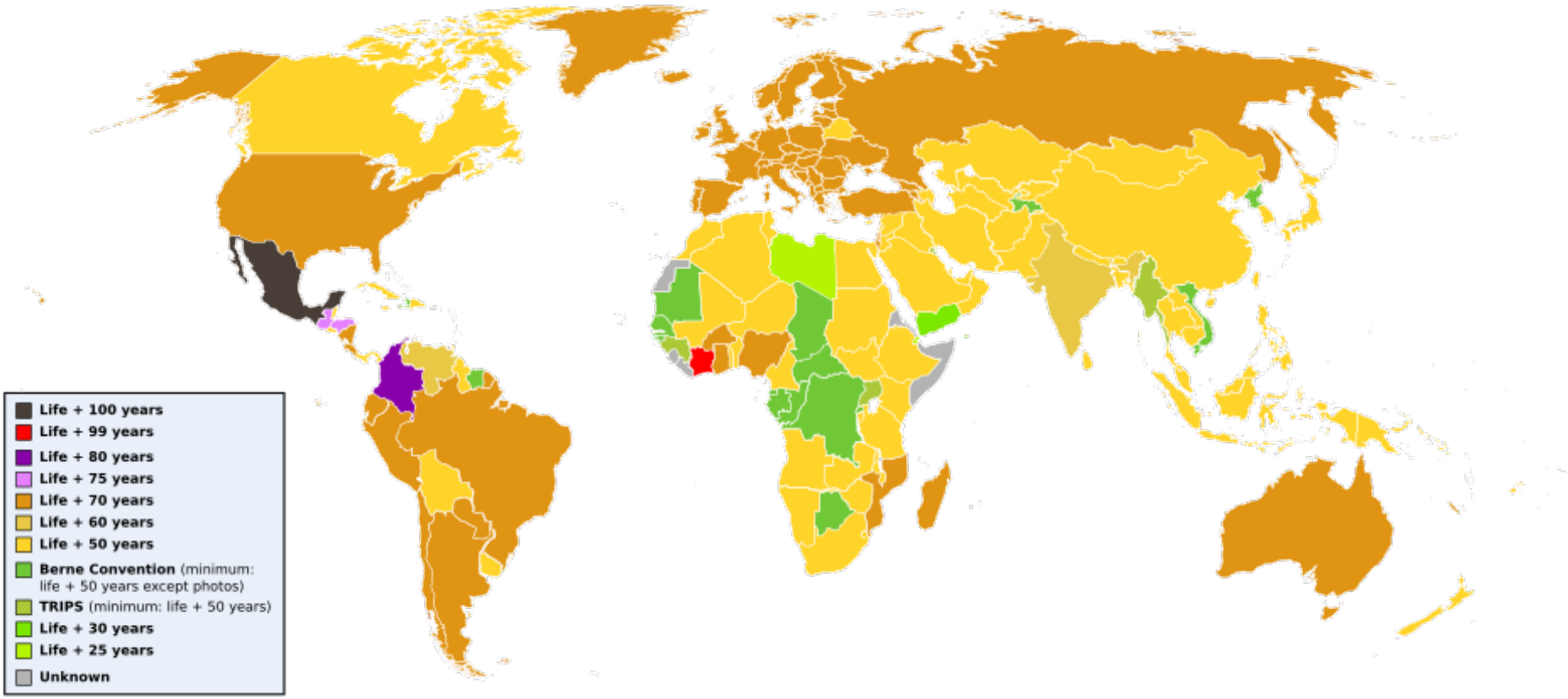
Public domain

Works in the public domain are those works for which copyright rights have expired, forfeited, or not applicable.

There are thus various ways for copyrightable works to enter the public domain:

1. copyright is not applicable to this kind of works (facts, theories, short snippets, . . .)
2. the copyright has expired
   - copyright expired *tout court*
     - see Public Domain Day, January 1st each year
   - the copyright owner failed to follow copyright renewal rules, where/when applicable (e.g., the novel *Anthem* by Ayn Rand)
3. the copyright owner *deliberately* places it in the public domain

World copyright terms



https://commons.wikimedia.org/wiki/File:World_copyright_terms.svg

Public domain dedication I

To deliberately put a work in the public domain, authors should use a public domain dedication, e.g.:

The person or persons who have associated work with this document (the "Dedicator" or "Certifier") hereby either (a) certifies that, to the best of his knowledge, the work of authorship identified is in the public domain of the country from which the work is published, or (b) hereby dedicates whatever copyright the dedicators holds in the work of authorship identified below (the "Work") to the public domain. A certifier, moreover, dedicates any copyright interest he may have in the associated work, and for these purposes, is described as a "dedicator" below.

A certifier has taken reasonable steps to verify the copyright status of this work. Certifier recognizes that his good faith efforts may not shield him from liability if in fact the work certified is not in the public domain.

Dedicator makes this dedication for the benefit of the public at large and to the detriment of the Dedicator's heirs and successors. Dedicator intends this dedication to be an overt act of relinquishment in perpetuity of all present and future rights under copyright law,

Public domain dedication II

whether vested or contingent, in the Work. Dedicator understands that such relinquishment of all rights includes the relinquishment of all rights to enforce (by lawsuit or otherwise) those copyrights in the Work.

Dedicator recognizes that, once placed in the public domain, the Work may be freely reproduced, distributed, transmitted, used, modified, built upon, or otherwise exploited by anyone for any purpose, commercial or non-commercial, and in any way, including by methods that have not yet been invented or conceived.

http://creativecommons.org/licenses/publicdomain/

Public domain applicability

- Is it actually possible *before* copyright expiration?

Public domain applicability

- Is it actually possible *before* copyright expiration?
- Several legal systems (and most notably in Europe) effectively prohibit any attempt by the owners to surrender copyright rights automatically conferred by law
  - Particularly moral rights (perpetual, unwaiverable, inalienable)
- A solution: the CC0 license by Creative Commons—waive all copyright rights to the fullest extent allowed by law

CC0 I

[...]

2. Waiver. To the greatest extent permitted by, but not in contravention of, applicable law, Affirmer hereby overtly, fully, permanently, irrevocably and unconditionally waives, abandons, and surrenders all of Affirmer's Copyright and Related Rights and associated claims and causes of action, whether now known or unknown (including existing as well as future claims and causes of action), in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "Waiver"). Affirmer makes the Waiver for the benefit of each member of the public at large and to the detriment of Affirmer's heirs and successors, fully intending that such Waiver shall not be subject to revocation, rescission, cancellation, termination, or any other legal or equitable action to disrupt the quiet enjoyment of the Work by the public as contemplated by Affirmer's express Statement of Purpose.

3. Public License Fallback. Should any part of the Waiver for any reason be judged legally invalid or ineffective under applicable law, then the

CC0 II

Waiver shall be preserved to the maximum extent permitted taking into account Affirmer's express Statement of Purpose. In addition, to the extent the Waiver is so judged Affirmer hereby grants to each affected person a royalty-free, non transferable, non sublicensable, non exclusive, irrevocable and unconditional license to exercise Affirmer's Copyright and Related Rights in the Work (i) in all territories worldwide, (ii) for the maximum duration provided by applicable law or treaty (including future time extensions), (iii) in any current or future medium and for any number of copies, and (iv) for any purpose whatsoever, including without limitation commercial, advertising or promotional purposes (the "License"). The License shall be deemed effective as of the date CC0 was applied by Affirmer to the Work. Should any part of the License for any reason be judged legally invalid or ineffective under applicable law, such partial invalidity or ineffectiveness shall not invalidate the remainder of the License, and in such case Affirmer hereby affirms that he or she will not (i) exercise any of his or her remaining Copyright and Related Rights in the Work or (ii) assert any associated claims and causes of action with respect to the Work, in either case contrary to Affirmer's express Statement of Purpose.

[...]

CC0 III

http://creativecommons.org/publicdomain/zero/1.0/

Outline

Mozilla Public License (MPL)

1998 version 1.0, as a successor of the NPL (Netscape Public License)

1999 version 1.1 by Mozilla Organization

- public feedback/review process on how to improve version 1.0
- allow for dual-/multi-licensing

2012 version 2.0

- public process again
- GPL-compatible

Mozilla Public License (MPL) 2.0

- weak (or partial) copyleft license, with file-based boundaries on the reach of copyleft requirements
- separation between covered software and larger work

  Covered Software

  > *means Source Code Form to which the initial Contributor has attached the notice in Exhibit A, the Executable Form of such Source Code Form, and Modifications of such Source Code Form, in each case including portions thereof*

  Larger Work

  > *means a work that combines Covered Software with other material, in a separate file or files, that is not Covered Software*

See `http://opensource.org/licenses/MPL-2.0`

Mozilla Public License (MPL) 2.0 — Exhibit A

Exhibit A - Source Code Form License Notice

This Source Code Form is subject to the terms of the Mozilla Public License, v. 2.0. If a copy of the MPL was not distributed with this file, You can obtain one at https://mozilla.org/MPL/2.0/.

i.e., license notice

Mozilla Public License (MPL) 2.0 (cont.)

Covered software

1. copyleft clause

   All distribution of Covered Software in Source Code Form, including any Modifications that You create or to which You contribute, must be under the terms of this License

2. source code requirement

   If You distribute Covered Software in Executable Form then [...] such Covered Software must also be made available in Source Code Form

Mozilla Public License (MPL) 2.0 (cont.)

- Larger work: its own license with carve out for MPL-covered code

  You may create and distribute a Larger Work under terms of Your choice, provided that You also comply with the requirements of this License for the Covered Software.

- explicit multiple-licensing support:

  If the Larger Work is a combination of Covered Software with a work governed by one or more Secondary Licenses, [. . .], this License permits You to additionally distribute such Covered Software under the terms of such Secondary License(s), so that the recipient of the Larger Work may, at their option, further distribute the Covered Software under the terms of either this License or such Secondary License(s).

Mozilla Public License (MPL) 2.0 (cont.)

- Patent provisions: patent license + patent retaliation (similar to Apache2)
- approved by: FSF, OSI, Debian
- version 2.0 of the license is compatible with the GPL
- version 1.1 is *in*compatible with the GPL
  - A module covered by the GPL and a module covered by the MPL version 1.1 cannot be linked together.
  - For this reason, Firefox has been relicensed under multiple licenses (MPL, GPL, LGPL).
  - MPL 1.1 can be specifically amended to allow combining with GPL and others (§13, “Multiple-licensed code”).

CDDL License

- The Common Development and Distribution License (CDDL) is based on the MPL, version 1.1
- Produced by Sun Microsystems for the OpenSolaris projects (kernel, userland, ZFS, DTrace, NetBeans, GlassFish, . . .)
- approved by: FSF, OSI, Debian
- It tries to amend GPL-incompatibility issues in the MPL 1.1
  - without succeeding (according to FSF and Debian)
- Some non-compliance issues with European law system in the MPL have been corrected in the CDDL
- approved by: FSF, OSI, Debian
- GPL-incompatible

GNU LGPL

1991 GNU *Library* General Public License, version 2 (for uniformity with GPL version)

1999 GNU *Lesser* General Public License, version 2.1

- name change to emphasize that it is inferior (from a copyleft POV) to the GPL, rather then the recommended variant of the GPL for software libraries

2007 GNU LGPL, version 3

- reimplemented as GPLv3 + additional permissions

- very popular license for libraries (and more)
- approved by: FSF, OSI, Debian
- GPL compatible

GNU LGPL 2.1

- ***§4.*** *You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, [. . .]*

GNU LGPL 2.1

- ***§4.*** *You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, [. . .]*
- ***§5.*** *A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.*

Note the lack of explicit file boundaries (contrary to, e.g., MPL)

GNU LGPL 2.1

- ***§4.*** *You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, [. . .]*
- ***§5.*** *A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.*

Note the lack of explicit file boundaries (contrary to, e.g., MPL)

- ***§3.*** *You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library [. . .] This option is useful when you wish to copy part of the code of the Library into a program that is not a library.*

GNU LGPL 3 — definitions

- *A "Combined Work" is a work produced by combining or linking an Application with the Library. The particular version of the Library with which the Combined Work was made is also called the "Linked Version".*

- *The "Minimal Corresponding Source" for a Combined Work means the Corresponding Source for the Combined Work, excluding any source code for portions of the Combined Work that, considered in isolation, are based on the Application, and not on the Linked Version*

- *The object code form of an Application may incorporate material from a header file that is part of the Library.*

GNU LGPL 3 — mechanism

- *You may convey a Combined Work under terms of your choice that, taken together, effectively do not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering [. . .]*
- *[provided that] you do one of the following:*

1. *Convey the Minimal Corresponding Source under the terms of this License, and the Corresponding Application Code in a form suitable for, and under terms that permit, the user to recombine or relink the Application with a modified version of the Linked Version [. . .]*
2. *Use a suitable shared library mechanism for linking with the Library. [. . .]*

things start to get quite technical for a legal document. . .

Intuition: enable users to enjoy the 4 freedoms on the "library". Technically, that means being able to replace the library with a modified version of it.

Outline

GNU General Public License (GPL)

- considered to be the most popular Free Software license
- approved by: FSF, Debian, OSI

1989 version 1 (by RMS), as a generalization (hence the name) of licenses already used by the GNU project for: Emacs, GDB, GCC

1991 version 2 (by RMS)

- "liberty or death"; early ex. of defense against patents and similar threats to user freedoms

2007 version 3 (by RMS with counsel from E. Moglen/SFLC)

- public review process
- software patents clauses
- DRM clauses (anti "tivoization")
- license compatibility provision
- internationalization
- self-defense against further restrictions

GPL — relevance

What makes the GPL so special?

- First implementation of copyleft
- Highly influential on all subsequent copyleft/share-alike licenses, including Creative Commons
- Without the GPL, copyleft would have been just an abstract idea
- Designed to prevent proprietary relicensing of Free Software code
- Popularity

GPLv2 — source code requirement

§3. *You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:*

a) *Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, [. . .]*

5. the so called "system library exception"

GPLv2 — source code requirement

§3. *You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:*

a) *Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, [. . .]*

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable.

5. the so called "system library exception"

GPLv2 — source code requirement

§3. *You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:*

a) *Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or, [. . .]*

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable.
However, as a special exception,[5] *the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.*

5. the so called "system library exception"

GPLv2 — copyleft

§2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) *You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.*

b) *You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.*

- derived works fall under the terms of the GPL themselves, hence *their* source code must be distributed as well
- (a) is a local requirement

  (b) only triggers upon distribution

GPLv2 — no EULA

§5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

GPLv2 — no EULA

> ***§5.*** *You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.*

How about permission to *use* the Program?

GPLv2 — no EULA

> ***§5.*** *You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.*

How about permission to *use* the Program?

> ***§0.*** *[...] Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, [...]*

GPLv2 — “or later”

Recommended way to apply the GPL to source code:

> *This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.*

- part of the copyright/license notices, not of the license itself
- individual software authors *can* leave the “or later” clause out
- other licenses include implicit “or later” requirements in the license text itself (e.g., MPL)

For best practices on how to manage copyright/license notices see: Software Freedom Law Center, *Managing copyright information within a free software project*

`https://softwarefreedom.org/resources/2012/ManagingCopyrightInformation.html`

LGPL vs GPL — rationale

Why you shouldn't use the Lesser GPL for your next library
https://www.gnu.org/licenses/why-not-lgpl.html

The GNU Project has two principal licenses to use for libraries. One is the GNU Lesser GPL; the other is the ordinary GNU GPL. [...] using the Lesser GPL permits use of the library in proprietary programs; using the ordinary GPL for a library makes it available only for free programs. [...]

Which license is best for a given library is a matter of strategy [...]. [...] Free software developers need to make advantages for each other. Using the ordinary GPL for a library gives free software developers an advantage over proprietary developers: a library that they can use, while proprietary developers cannot use it. [...]

There are reasons that can make it better to use the Lesser GPL in certain cases. The most common case is when a free library's features are readily available for proprietary software through other alternative libraries. In that case, the library cannot give free software any particular advantage, so it is better to use the Lesser GPL for that library.

GPLv2 — looking back

- Written by Richard Stallman and the FSF, published in 1991.
- The most popular Free Software license: estimated to cover 50-70 % of all Free Software projects (at the time)
- It's more than a software license: it is a social contract, imposing that all players have the same rights and obligations

Why update it?

GPLv2 — looking back

- Written by Richard Stallman and the FSF, published in 1991.
- The most popular Free Software license: estimated to cover 50-70 % of all Free Software projects (at the time)
- It's more than a software license: it is a social contract, imposing that all players have the same rights and obligations

Why update it?
After 15 years, needed updating in order to remain effective against new threats to user freedoms.

Intuition: the GPL is a mean to an end. It is an implementation that might have bugs (or grow them over time), which need to be fixed in further releases of the license.

GPLv3 — public consultation

Public consultation process:

- very relevant and the social responsible thing to do: given the abundance of “or later” software, the effects of the release of GPLv3 might be huge
- It lasted 18 months: from January 16, 2006 (first draft) to June 29, 2007 (final version)
- Invited participants from high-profile Free Software projects
- 4 drafts
- 5 International Conferences (Boston, Porto Alegre, Barcelona, Tokyo and Brussels)

GPLv3 — DRM

> ***§3. Protecting Users' Legal Rights From Anti-Circumvention Law.***
>
> *[. . .]*
>
> *When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.*

- does not *forbid* to implement DRM & co. in software
- but allows to write interoperable software and bypass restrictions
- neutralize laws that get in the way of user freedoms (e.g., DMCA, EUCD)

GPLv3 — DRM (cont.)

- Together with

  §6. Conveying Non-Source Forms.

  [. . .]

  "Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source.

  it also neutralizes "tivoization", i.e., the circumvention of the GPL by using cryptography to disallow the installation/execution of modified versions of a GPL'd program

GPLv2 — patents

Protection against patent threats is implemented by GPLv2 only in the "Liberty or Death" clause:

> ***§7.*** *If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all.*
>
> *[. . .]*
>
> *It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. [. . .]*

GPLv3 — patents

GPLv3 *adds* (i.e., "liberty or death" remains) stronger protection against patent threats through legal-engineering:

> ***§11. Patents.***
>
> *[. . .] Each contributor grants you a non-exclusive, world-wide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.*

> ***§10. Automatic Licensing of Downstream Recipients.***
>
> *[. . .] you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.*

GPLv3 — self-defense

§7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. [. . .]

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. [. . .]

All other non-permissive additional terms are considered "further restrictions" [. . .] If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term.

GPLv3 — variants and compatibility

- ***§7. Additional Terms.***
  [. . .] for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms: [. . .]
  e) *Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or*

  (and other similar permissions for adding warranties/"as is" disclaimers, limiting the use for publicity purposes, etc.)

- *Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work.*

GPLv3 — Warranty and Liability

§15. Disclaimer of Warranty.

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

§16. Limitation of Liability.

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

The Application Service Provider (ASP) loophole

Exercise

1. *obtain a copy of some GPL'd program*
2. *modify it*
3. *offer remote access to your modified version over the Net (e.g., web app, remote API, etc.)*

Does the GPL force you to redistribute the code of your modified version?

The Application Service Provider (ASP) loophole

Exercise

1. *obtain a copy of some GPL'd program*
2. *modify it*
3. *offer remote access to your modified version over the Net (e.g., web app, remote API, etc.)*

Does the GPL force you to redistribute the code of your modified version? ***No.***

- GPL (both v2 and v3) copyleft clauses trigger upon distribution of the modified copy, in either source or binary form
- if you do not do any of that, copyleft does not kick in
- from copyleft POV, this is very problematic for web/network apps

  "GPL is the BSD of Web applications" — *Bradley Kuhn*

- but in an increasingly more connected world, the problem is more general

GNU Affero GPL (AGPL)

- Based on the GPL
- Pioneered by Affero, Inc. (version 1, 2002)
- Published by the Free Software Foundation (version 3, 2007)
- It contains the extra Affero clause that requires distribution of modified source code of applications to users interacting remotely over the network with the program
- The clause has initially been considered for inclusion in GPLv3, but then relegated to a separate license
- approved by: FSF, Debian, OSI
- GPL compatible (explicitly so)

AGPL — Affero clause

§13. Remote Network Interaction *[. . .]*

if you modify the Program, your modified version must prominently offer all users interacting with it remotely through a computer network (if your version supports such interaction) an opportunity to receive the Corresponding Source of your version by providing access to the Corresponding Source from a network server at no charge, through some standard or customary means of facilitating copying of software.

Outline

Outline

Derivative works and the GPL

GPL copyleft propagation applies to (GPLv2 language):

> *a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.*

Exercise (linking and the GPL)

Can you link a GPL program/library with a non-GPL program/library, without applying the GPL to the obtained binary?

6. according to some, the *actual* answer is thus "we don't know"

Derivative works and the GPL

GPL copyleft propagation applies to (GPLv2 language):

> *a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.*

Exercise (linking and the GPL)

Can you link a GPL program/library with a non-GPL program/library, without applying the GPL to the obtained binary?

Answer: [6]

- FSF (and popular) answer: no; the GPL applies
- some corporate lawyers' answer: yes; the GPL doesn't apply
- court cases/tribunal answer: none (yet)

6. according to some, the *actual* answer is thus "we don't know"

Derivative or collective works?

[US law language]

- a derivative work is a "work based upon one or more preexisting works", which requires some transformation or adaption of the original
- a collective work is created when a person brings together "preexisting materials. . . in such a way that the resulting work as a whole constitutes an original work of authorship"
  - individual parts remain under their individual licenses
  - a separate license apply to the collection

Does linking create a derivative or a collective work (or both)?

Linking and the GPL — FSF position

License text (redux):

> *a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.*

From the GPL FAQ: [7]

> *Q: Does the GPL have different requirements for statically vs dynamically linked modules with a covered work?*
> *A: No. Linking a GPL covered work statically or dynamically with other modules is making a combined work based on the GPL covered work. Thus, the terms and conditions of the GNU General Public License cover the whole combination. [...]*
> *Q: Can I release a non-free program that's designed to load a GPL-covered plug-in?*
> *A: [...] Using shared memory to communicate with complex data structures is pretty much equivalent to dynamic linking*

7. http://www.gnu.org/licenses/gpl-faq.html

Linking and the GPL — arguments

- dynamically linked executables contains “annotations and elaborations” on a base binary
  - does a Linux kernel module contains annotations and elaborations on the base expression of the kernel?
  - if yes, then it might be a derived work of the kernel (GPLv2)

8. https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/COPYING

Linking and the GPL — arguments

- dynamically linked executables contains “annotations and elaborations” on a base binary
  - does a Linux kernel module contains annotations and elaborations on the base expression of the kernel?
  - if yes, then it might be a derived work of the kernel (GPLv2)
  - how about user programs that run on Linux?
  - according to Linus and kernel developers: [8]

*NOTE! This copyright does *not* cover user programs that use kernel services by normal system calls - this is merely considered normal use of the kernel, and does *not* fall under the heading of “derived work”.*

8. https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/COPYING

Linking and the GPL — arguments

- dynamically linked executables contains “annotations and elaborations” on a base binary
  - does a Linux kernel module contains annotations and elaborations on the base expression of the kernel?
  - if yes, then it might be a derived work of the kernel (GPLv2)
  - how about user programs that run on Linux?
  - according to Linus and kernel developers: [8]

  *NOTE! This copyright does *not* cover user programs that use kernel services by normal system calls - this is merely considered normal use of the kernel, and does *not* fall under the heading of “derived work”.*

  . . . but are they right?
- the legal principle of usage of trade might play a role too

8. https://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/tree/COPYING

Linking and the GPL — arguments (cont.)

- arguments to the contrary (often by corporate lawyers) claim that linking only creates collective works—not subject to the GPL as a whole—because there is no substantial difference between two executables on disk and two in memory

Linking and the GPL — arguments (cont.)

- arguments to the contrary (often by corporate lawyers) claim that linking only creates collective works—not subject to the GPL as a whole—because there is no substantial difference between two executables on disk and two in memory
- header files might also play a role
  - during compilation (before linking) you might use header files to prepare your executable for dynamic linking
  - if the headers used at compile time are GPL'd, then your dynamically linked executable might be a derived work of the headers

Linking and the GPL — arguments (cont.)

- arguments to the contrary (often by corporate lawyers) claim that linking only creates collective works—not subject to the GPL as a whole—because there is no substantial difference between two executables on disk and two in memory
- header files might also play a role
  - during compilation (before linking) you might use header files to prepare your executable for dynamic linking
  - if the headers used at compile time are GPL'd, then your dynamically linked executable might be a derived work of the headers

Exercise (LGPL header files v. linking)

Is header inclusion a potential issue for LGPL'd programs too? Why?

Linking and the GPL — arguments (cont.)

- there are also other types of “linking”: RPC, RMI, REST API, etc. When do they constitute “linking” in a sense that would trigger strong copyleft requirements?
  - no consensus yet
  - legal folklore seems to suggest that:
    - loosely coupled and/or popular and/or standardized APIs with several alternative implementations should not trigger the GPL
    - tightly coupled and/or ad-hoc and/or single-implementation APIs should trigger the GPL
- on the other hand, it seems consensual that static linking produces a derivative work of the GPL part

Linking and the GPL — arguments (cont.)

- there are also other types of "linking": RPC, RMI, REST API, etc. When do they constitute "linking" in a sense that would trigger strong copyleft requirements?
  - no consensus yet
  - legal folklore seems to suggest that:
    - loosely coupled and/or popular and/or standardized APIs with several alternative implementations should not trigger the GPL
    - tightly coupled and/or ad-hoc and/or single-implementation APIs should trigger the GPL
- on the other hand, it seems consensual that static linking produces a derivative work of the GPL part

Ultimately, this GPL linking dilemma is problematic only for those who want to somehow circumvent one of the main goals of the GPL which, *per se*, is very clear.

Outline

CAA/CLA

Copyright Assignment Agreement (CAA) cession agreement where a copyright holder surrender all their copyright sanctioned rights on some work to another party

Contribution License Agreement (CLA) agreement where a copyright holder gives a license (usually non-revocable, possibly exclusive) to enforce specific copyright sanctioned rights to another party

- on paper, CAA are more powerful than CLA; but they only go as far as the legal system allows them
  - e.g., in most of Europe moral rights cannot be surrendered
- CLAs can be so broad to be *de facto* equivalent to CAAs
- key copyright right for policy reasons: the ability to relicense

If a vendor participating in a FOSS project has, alone, the ability to relicense, strategy considerations based solely on the chosen FOSS license are completely moot.

Not all CAAs/CLAs are born equal

- set of rights surrendered
  - e.g., enforcement-only agreements
- mandatory vs optional agreements
- to (public benefit) nonprofit vs for profit entities
- safe guards
  - e.g., “we can relicense, but we will pick within this set of licenses”
    - common choice: OSI-approved ∩ FSF-approved licenses
- alternatives (within limits)
  - “or later” clauses
    - possibly with license proxy (e.g., GPLv3, §14)
  - will

Outline

Warning

Due to the free circulation of Free Software, it is very difficult to get hard number about software—and therefore license—popularity.

Nonetheless, many actors of the FOSS ecosystem publish statistics about those facts. Unfortunately, most of them do so in rather unscientific ways: without disclosing the details about the dataset they are using, and without liberating the software they use to compile their statistics.

Use caution in interpreting the data.

Blackduck: “Top 20 Open Source Licenses”

GPL 2.0	25%
MIT	19%
Apache License 2.0	16%
GPL 3.0	10%
BSD 3-clause	7%
Artistic (Perl)	5%
LGPL 2.1	5%
LGPL 3.0	2%
MS-PL	2%
EPL	2%
Code Project Open License	1%
MPL 1.1, BSD 2-clause, CDDL 1.0, AGPL, Microsoft Reciprocal License, Sun GPL w/ classpath exc., CDDL 1.1, zlib/libpng, CPL	<1%[9]

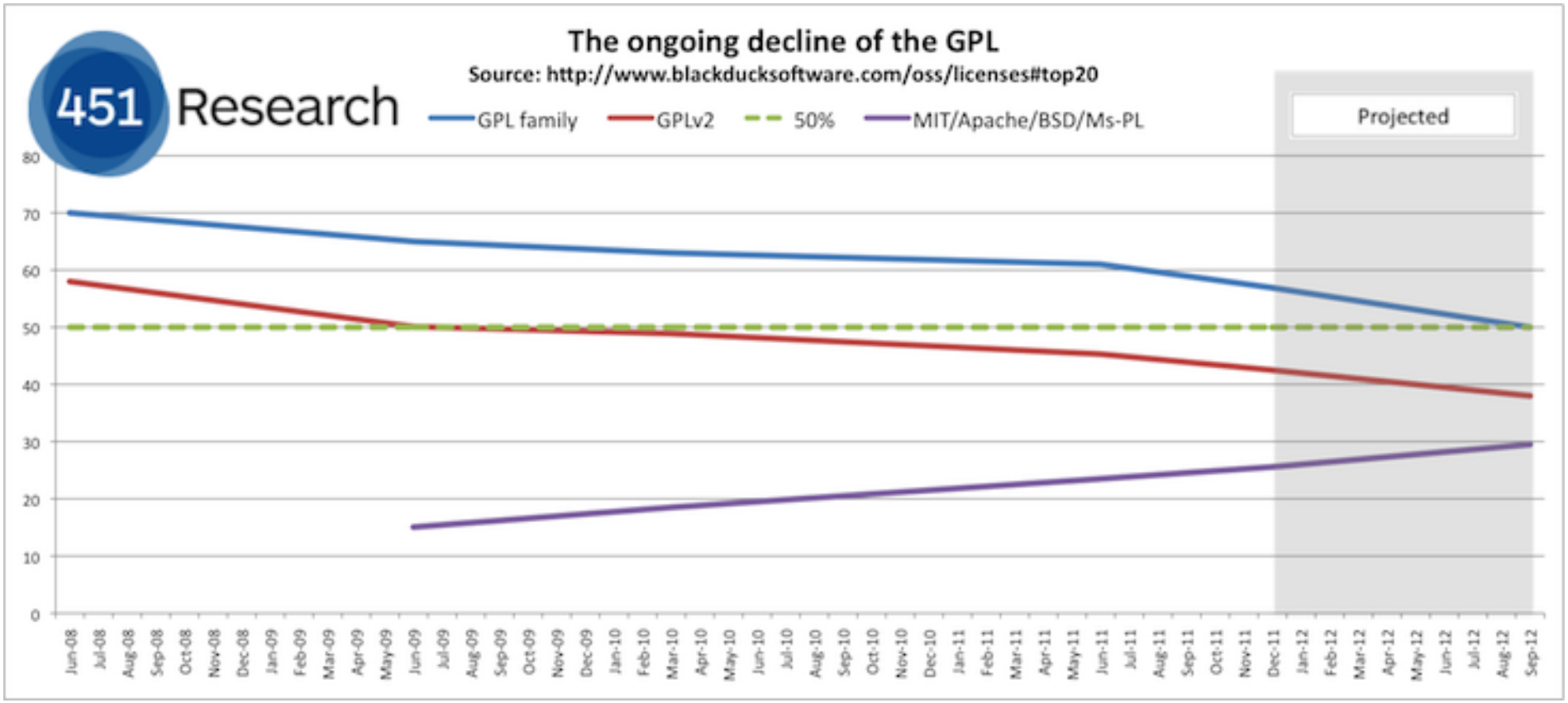
https://www.blackducksoftware.com/resources/data/top-20-open-source-licenses

February 2015

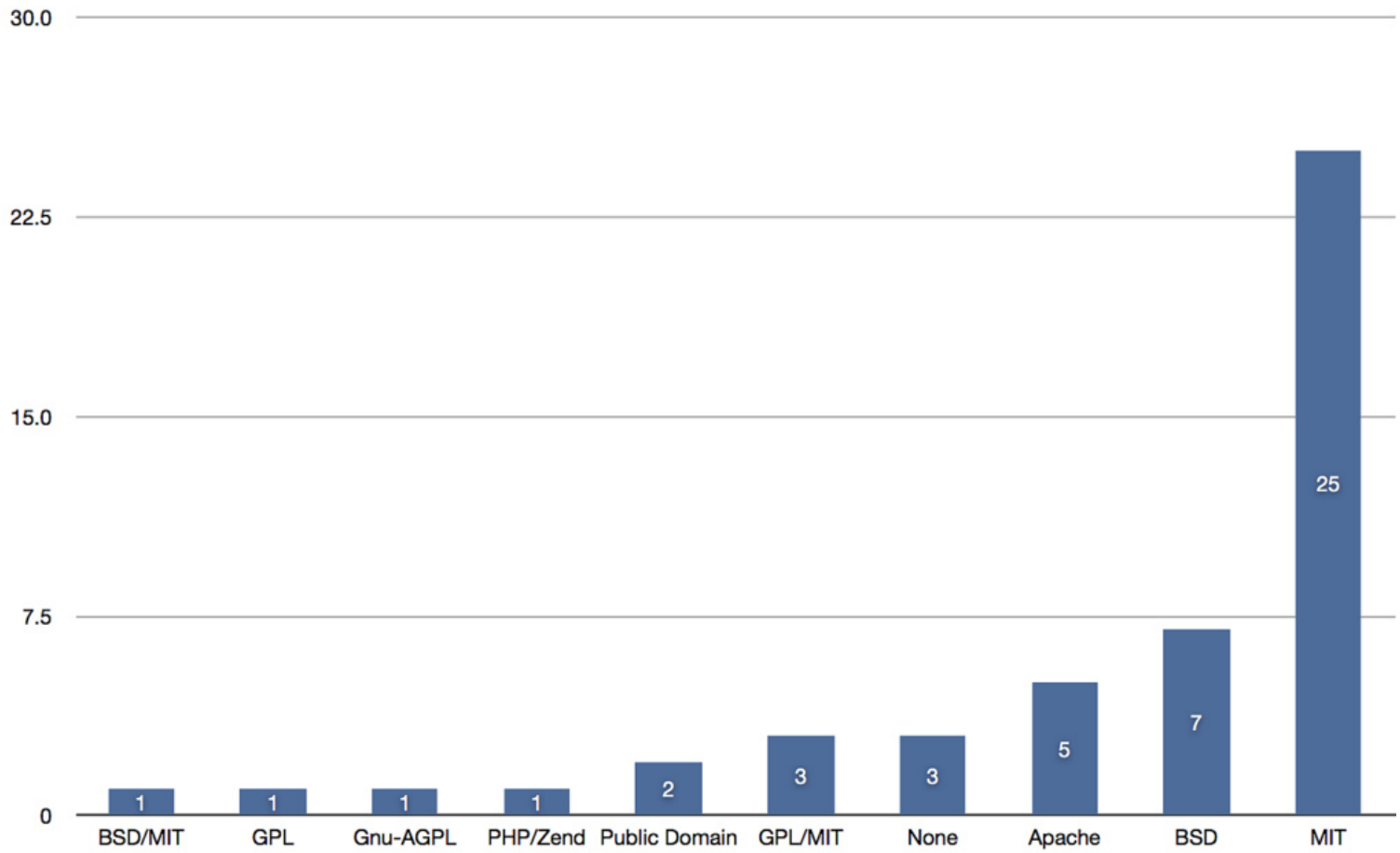
9. each license

Blackduck: the (alleged) decline of the GPL

According to Blackduck, in recent years the GPL is losing ground to permissive licenses:



GitHub: top licenses



http://ostatic.com/blog/the-top-licenses-on-github

February 2012

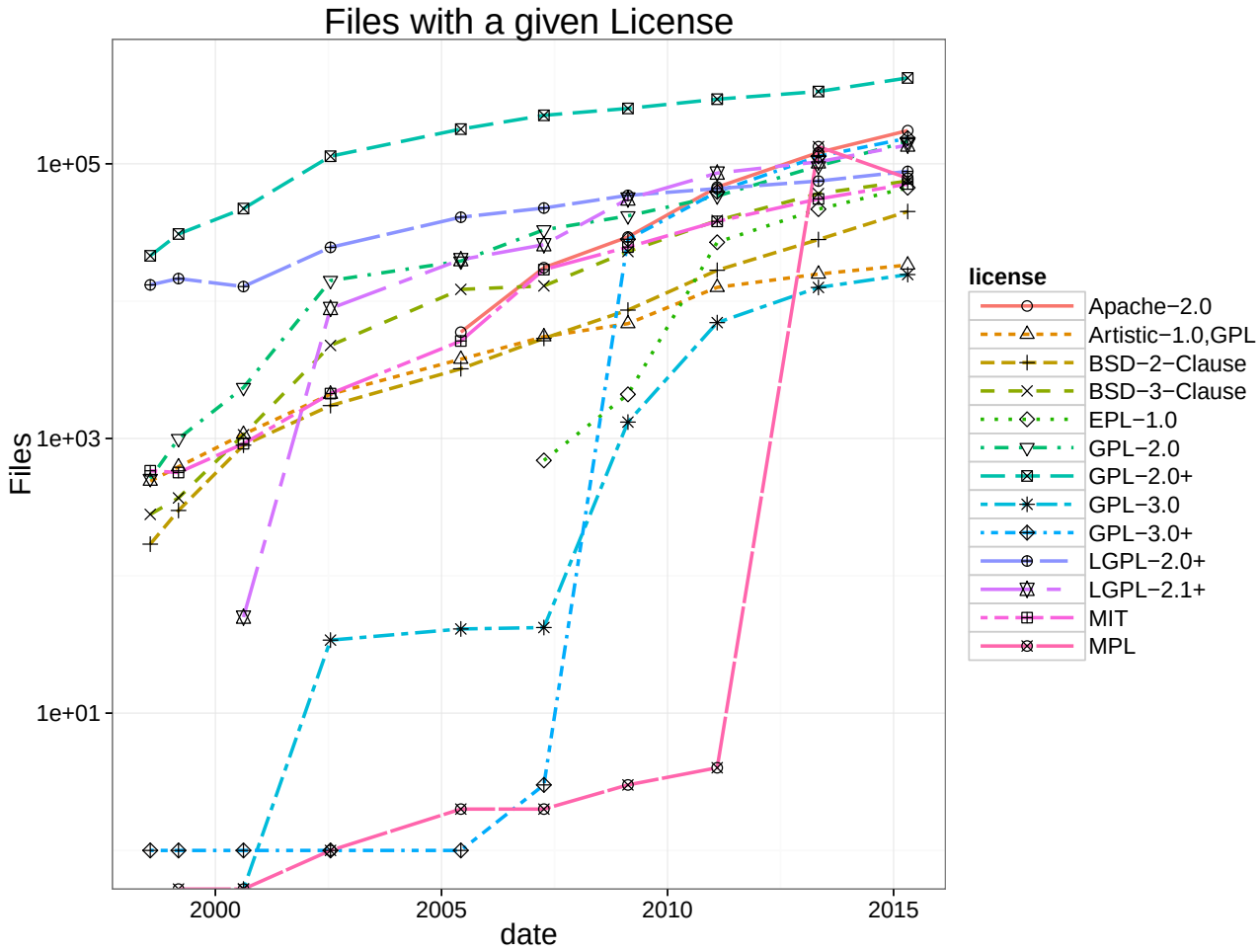
GitHub — a bit more complex than that

- 2013 analysis of GitHub by Aaron Williamson "Licensing of Software on Github: A Quantitative Analysis"[10]
- only 14.9% have a top-level license file
- only 3.6% mention a license in README
- for the remaining projects, the breakdown is largely confirmed
- POSS ("Post Open Source Software" debate)

10. http://www.softwarefreedom.org/resources/2013/lcs-slides-aaronw/

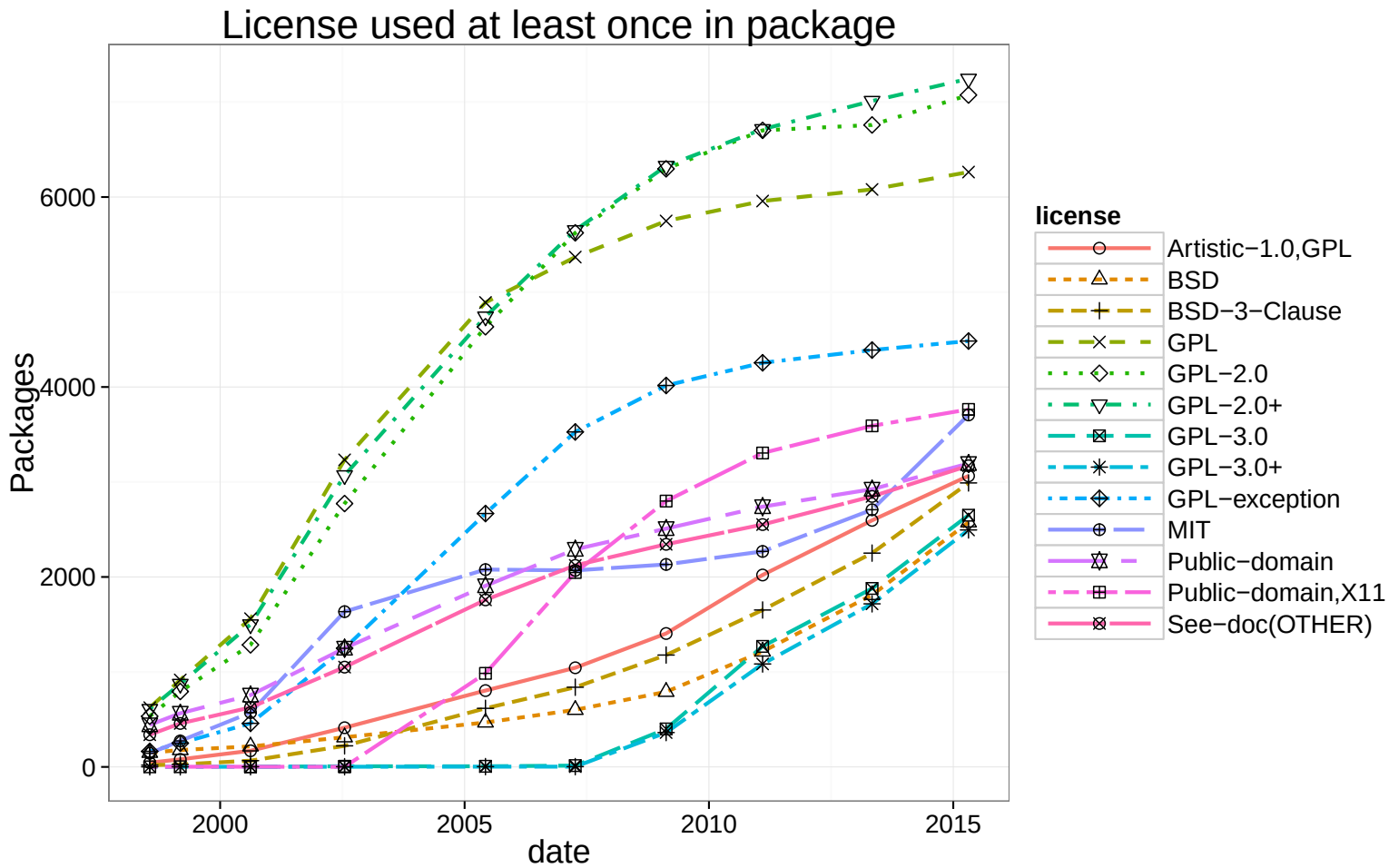
License popularity in Debian over time

Matthieu Caneill, Daniel M. Germán, Stefano Zacchiroli
The Debsources Dataset: Two Decades of Free and Open Source Software.
Empirical Software Engineering, Vol. 22, 2017.



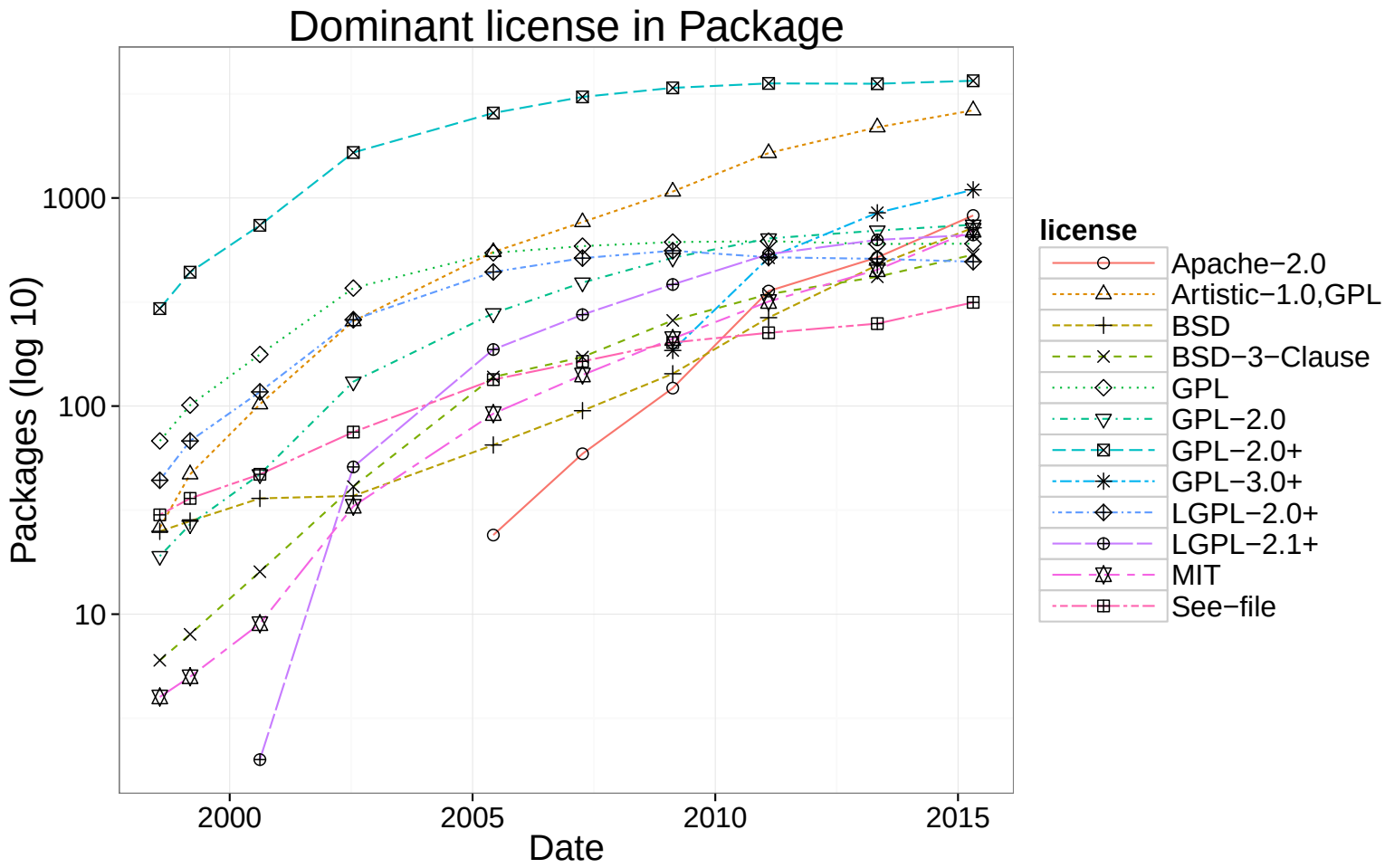
License popularity in Debian over time (cont.)

Matthieu Caneill, Daniel M. Germán, Stefano Zacchiroli
The Debsources Dataset: Two Decades of Free and Open Source Software.
Empirical Software Engineering, Vol. 22, 2017.

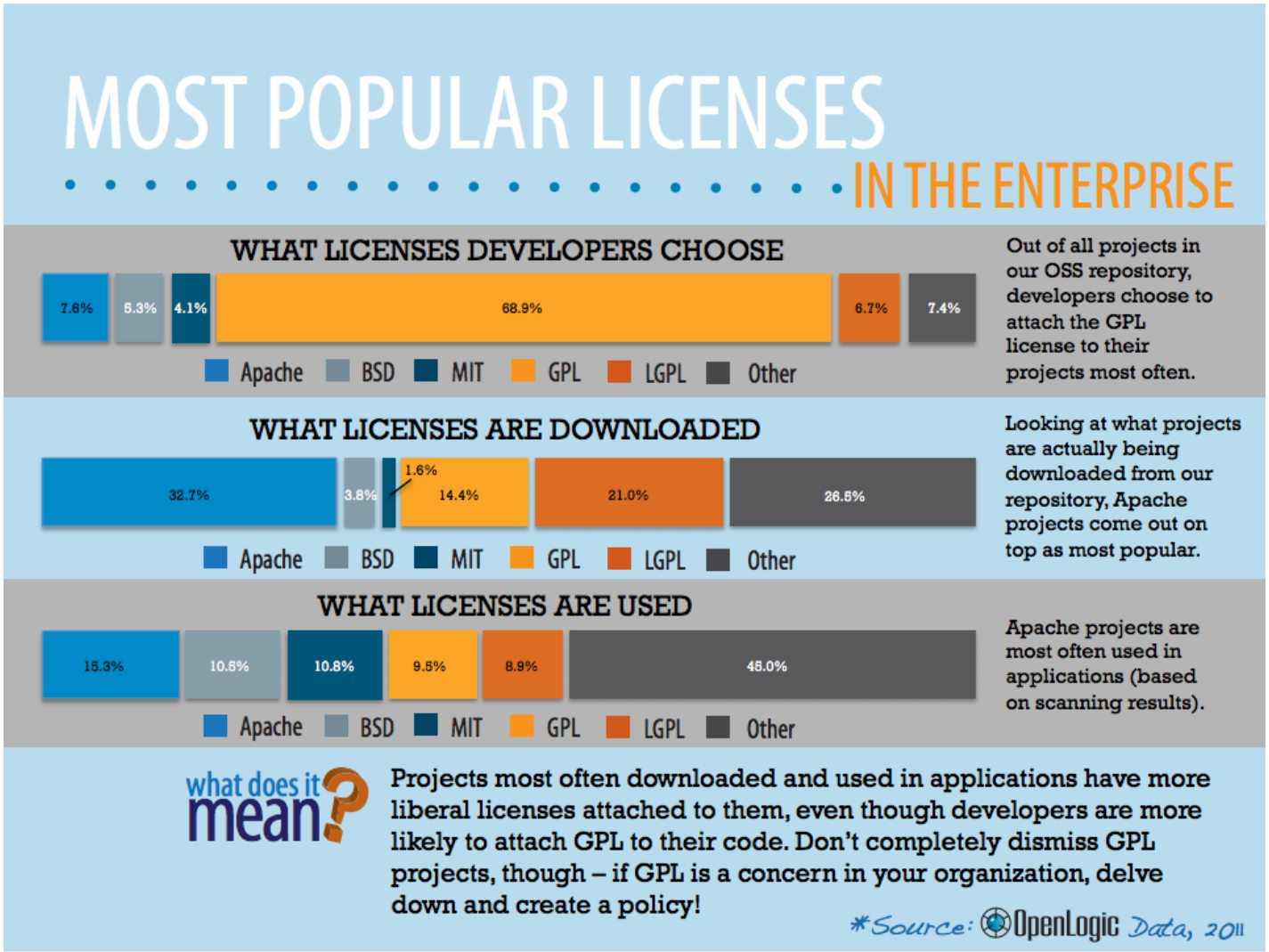


License popularity in Debian over time (cont.)

Matthieu Caneill, Daniel M. Germán, Stefano Zacchiroli
The Debsources Dataset: Two Decades of Free and Open Source Software.
Empirical Software Engineering, Vol. 22, 2017.



OpenLogic: most popular licenses in the enterpises



http://www.openlogic.com/blog/bid/197148/open-source-software-101-understanding-compliance

July 2012

Discussion: GPL vs BSD

- “BSD code is free, but GPL code stays free”
- copyleft: (non) business friendly?

Outline

Free riding

Definition (Free-rider problem)

The free-rider problem occurs when those who benefit from resources, public goods, or services do not pay for them, which results in an underprovision of those goods or services

FOSS is at risk of free-riding, in particular when developers are not sufficiently funded and not enough users contribute back to address developers need.

Free riding

Definition (Free-rider problem)

The free-rider problem occurs when those who benefit from resources, public goods, or services do not pay for them, which results in an underprovision of those goods or services

FOSS is at risk of free-riding, in particular when developers are not sufficiently funded and not enough users contribute back to address developers need.

Large “cloud” providers are prone to FOSS free riding, e.g.:

- take an existing FOSS component, e.g., a server-side DB
- integrate it so that cloud users can automatically deploy it, e.g., in a VM
- charge cloud users for the use of the service and/or the VM, e.g., by uptime, no. of requests, etc.

Anti free-riding licenses

A number of anti free-riding licenses are being experimented with to address the problem.

The balance between addressing the issue and retaining FOSS freedoms (and in particular freedom 0) is hard to achieve though.

To date, no such license has been recognized as free software and/or open source.

Commons Clause

> *Without limiting other conditions in the License, the grant of rights under the License will not include, and the License does not grant to you, the right to Sell the Software.*
> *[. . .] "Sell" means practicing any or all of the rights granted to you under the License to provide to third parties, for a fee or other consideration (including without limitation fees for hosting or consulting/support services related to the Software), a product or service whose value derives, entirely or substantially, from the functionality of the Software.*
>
> — *https://commonsclause.com*

- clause meant to be added to other licenses, in particular (but not only) Apache
- does not claim to be FOSS; from the FAQ: "Q: is this open source?" "A: No."
- adoption: extensions for Redis, see https://redislabs.com/community/licenses/

Server-Side Public License I

Based on AGPL, replacing §13 with the following:

> *§13. Offering the Program as a Service.*
>
> *If you make the functionality of the Program or a modified version available to third parties as a service, you must make the Service Source Code available via network download to everyone at no charge, under the terms of this License. Making the functionality of the Program or modified version available to third parties as a service includes, without limitation, enabling third parties to interact with the functionality of the Program or modified version remotely through a computer network, offering a service the value of which entirely or primarily derives from the value of the Program or modified version, or offering a service that accomplishes for users the primary purpose of the Program or modified version.*

Server-Side Public License II

"Service Source Code" means the Corresponding Source for the Program or the modified version, and the Corresponding Source for all programs that you use to make the Program or modified version available as a service, including, without limitation, management software, user interfaces, application program interfaces, automation software, monitoring software, backup software, storage software and hosting software, all such that a user could run an instance of the service using the Service Source Code you make available.

- `https://www.mongodb.com/licensing/server-side-public-license`
- adoption: MongoDB
- license submitted to OSI for approval, not accepted yet, mostly negative feedback thus far (February 2018)