Open Source software and dual licensing

Legal background of dual licensing

Caren Kresse
Open Source Automation Development Lab (OSADL) eG

Copyright law and dual licensing

- Copyright law grants the creator of a work extensive rights to decide **under which conditions** their work may be used, copied and distributed.
- A contract from the creator (or right holder) of a work that **gives permissions** of use and **imposes obligations** is called a **"license"**.
- A right holder may grant universal licenses to everyone or negotiate as many individual licenses as desired.
- The typical case of granting **two different** universal licenses is called **"dual licensing"**.

Proprietary vs. FOSS

Action	Proprietary software		FOSS	
	What is needed?	By whom/ why?	What is needed?	By whom/ why?
Run the software		**Prohibited** by the authors		**Permitted** by unilateral declaration of will of the authors
Analyze the software				
Modify the software				

Proprietary vs. FOSS

Action	Proprietary software		FOSS	
	What is needed?	By whom/ why?	What is needed?	By whom/ why?
Run the software	End User License Agreement (EULA)	**Permitted** by EULA	*Access to the software*	**Permitted** by unilateral declaration of will of the authors
Analyze the software		**Prohibited** by the authors	*Access to the software*	
Modify the software			*Access to the software*	

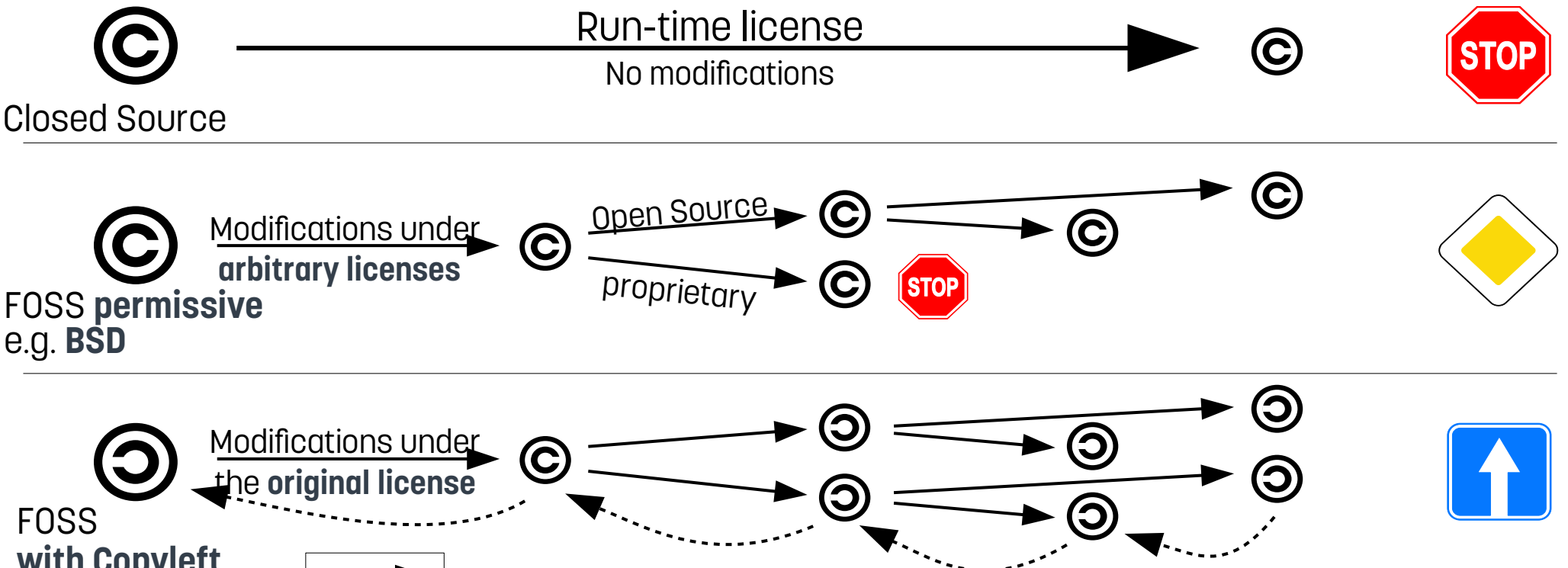
Proprietary vs. FOSS

Action	Proprietary software		FOSS	
	What is needed?	By whom/ why?	What is needed?	By whom/ why?
Run the software	End User License Agreement (EULA)	**Permitted** by EULA	*Access to the software*	**Permitted** by unilateral declaration of will of the authors
Analyze the software		**Prohibited** by the authors	*Access to the software*	
Modify the software			*Access to the software*	
Copy and distribute unmodified software		**Prohibited** by copyright law		**Prohibited** by copyright law
Copy and distribute modified software				

Proprietary vs. FOSS

Action	Proprietary software		FOSS	
	What is needed?	By whom/ why?	What is needed?	By whom/ why?
Run the software	End User License Agreement (EULA)	**Permitted** by EULA	*Access to the software*	**Permitted** by unilateral declaration of will of the authors
Analyze the software		**Prohibited** by the authors	*Access to the software*	
Modify the software			*Access to the software*	
Copy and distribute unmodified software	**License**	**Permitted** by license	**License**	**Permitted** by license
Copy and distribute modified software		**Prohibited** by copyright law	**License**	

What is “Copyleft”?



How does dual licensing work? (Example 1)

- Two FOSS licenses are given as alternatives, e.g.

```
/*
* Licensed under the Apache License 2.0 (the "License"). You may not use
* this file except in compliance with the License. You can obtain a copy
* in the file LICENSE in the source distribution or at
* https://www.openssl.org/source/license.html
*
* This file is dual-licensed and is also available under the following
* terms:
[...]
* 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.
[...]
*/
```

Apache-2.0

BSD-2-Clause

How does dual licensing work? (Example 2)

- Several FOSS licenses are given as alternatives, but one of them is specified for use in a particular context, e.g.

```
# Copyright (c) 2008 Andy Polyakov <appro@openssl.org>
#
# This module may be used under the terms of either the GNU General
# Public License version 2 or later, the GNU Lesser General Public
# License version 2.1 or later, the Mozilla Public License version
# 1.1 or the BSD License. The exact terms of either license are
# distributed along with this module. For further details see
# http://www.openssl.org/~appro/camellia/.
```

GPL-2.0-or-later,
LGPL-2.1-or-later,
MPL-1.1,
BSD

The website under the named link contains the information:

"In OpenSSL context, usage is effectively governed by the BSD License."

and the BSD-2-Clause license text.

BSD-2-Clause

How does dual licensing work? (Example 3)

- A proprietary licensing model is combined with a FOSS alternative, e.g.

  Oracle makes its MySQL database server and MySQL Client Libraries available under both the GPL and a commercial license. As a result, developers who use or distribute open source applications under the GPL can use the GPL-licensed MySQL software, and OEMs, ISVs and VARs that do not want to combine or distribute the MySQL software with their own commercial software under a GPL license can purchase a commercial license.

  GPL-2.0 WITH Universal-FOSS-exception-1.0

  Proprietary license

Why dual licensing?

There are various reasons for dual licensing an entire project under a proprietary license and a FOSS license.

- **For propagation:** A proprietary project wants to make particular parts or implementations available to the Open Source community.
- **For marketing:** A proprietary project wants to increase their reputation by calling themselves “Open Source”.
- **For funding:** A FOSS project cannot generate enough funding through foundations or services offered.
- etc.

The propagation reason

A proprietary project wants to make particular parts or implementations available to the Open Source community.

- Components that are **originally only licensed under a proprietary license** are alternatively made available under a FOSS license.
- This enables use for research or promotes a wider spread of the project.
- Example:
  NVIDIA Linux Open GPU Kernel Module, proprietary and MIT and GPL-2.0

The marketing reason

A proprietary project wants to increase their reputation by calling themselves “Open Source”.

- Components, especially **libraries**, are released by a company under **a proprietary license and a FOSS license with strong Copyleft** (usually GPL).
- This enables the company to call their product “Open Source”, although it is **not suitable for use in commercial products** as linking proprietary programs is not possible.
- This practice has led to a lot of **misunderstanding** of GPL and Open Source.
- Example:
  .NET StandardLibrary (OPC UA stack by the OPC Foundation), GPL-2.0 and RCL-1.0 only for OPC Foundation Corporate Members

The funding reason

A FOSS project cannot generate enough funding through foundations or services offered.

- A typical Open Source business model is based on providing the software for free and offering services around the software for a fee.
- If this does not generate enough funds, some projects are adopted and funded by foundations, companies or other organizations.
- If neither of these options are available to a FOSS project, they may make the decision to release **updates (or additional features) first under a proprietary license and only later (or not at all) under the FOSS license**.
- Example:
  MariaDB, proprietary and Business Source License (automatic relicenses to GPL-2.0-or-later after three years)

Contribution to dual licensed projects

- Successful FOSS projects rely on third-party contributions from the community.
- These ensure that the project develops in the direction required by users, that the quality remains high and that a stable community is formed.
- For dual licensed projects, **external contributions** can be organized in two ways:
  - Contributor License Agreement (CLA)
  - Internal firewall

Contributor License Agreement (CLA)

- Any contributor must sign a CLA that grants the right to **relicense the contribution under the proprietary license**, e.g.:

  "You acknowledge and agree that we may distribute the Contribution and Materials under the terms of any license approved by the Open Source Initiative or any other license approved by Us.", OPC Foundation Contributor Agreement
- Waiving own rights in such a far reaching manner, can discourage contributions from companies and individuals alike.

Internal firewall

- If no CLA is signed, the company **does not have the right to relicense** third-party code (under a Copyleft license) under the proprietary license.
- They must make sure that the contribution is not "accidentally" integrated into the proprietary version.
- If there is no sufficient internal separation, it may happen that **contributed FOSS code is merged into the proprietary version under violation of copyright**.
- If separation is good enough and there are many third-party contributions, it may happen that **two different versions** of the project develop.

Overview and summary of dual licensing

Advantages

- Flexible licensing models allow companies to access more diverse markets.
- Proprietary options can help fund development of FOSS projects.

Disadvantages

- Can be intentionally misused for marketing.
- Lack of understanding of how FOSS works can lead to insecurity about license choice.
- External contributions are discouraged.

→ Dual licensing is neither only good nor only bad, but it depends on the motivation and execution if the project remains usable.