OPEN SOURCE LICENSE COMPLIANCE

Richard E. Fontana
Open Source Licensing &
Patent Counsel, Red Hat
May 27, 2010

open source .com

AGENDA

- Historical background, definitions and characteristics
- License categories: copyleft (strong, weak), permissive
- License enforcement
- License compliance – due diligence; source code analysis; some mechanics

HISTORICAL BACKGROUND

- Nonproprietary code-sharing commons
- Exclusive property concepts gradually got mapped to software (©, trade secrets, patents)
- Gave rise to business models based on contractually licensing subsets of rights
- **Free software** licensing models emerged shortly thereafter – deploying legal machinery of restrictive licenses to encourage collaborative development, distributed improvement & widespread adoption

DEFINING FREE/LIBRE/OPEN SOURCE

- Hundreds of licenses customarily considered FLOSS
  - Newer projects standardizing around small set of popular licenses
- No single canonical definition
  - Evolving legal norms based in community consensus, embodied in development and distribution practices
  - Influential organizations: FSF, Debian, OSI, Fedora
- Everyone should adopt strictest community standards for what is/isn't authentic FLOSS

CHARACTERISTICS OF FLOSS

- Legal:
  - User gets a broad © license: perpetual, RF;
    - Essentially unlimited private use
    - Public use restricted only in ways not customarily regarded as **unduly burdensome** to software freedom
- Technical:
  - Either it's source code, or license provides for readily available source code at no further cost

LICENSE CATEGORIES

- Copyleft
  - Strong
  - Weak
- Permissive/Non-copyleft

open source .com

COPYLEFT

- License limits freedom of user to distribute derivative work under more restrictive terms
- Usually there is some source code disclosure requirement
- Typically, that source code, at least, must be under the same license as upstream

STRONG COPYLEFT (GPL)

- GPLv2 by far the most widely-used FLOSS license, for established as well as new projects
- Policy goal: Preserve free software commons, even as software gets improved downstream
- “Strong”: licensor expectation that copyleft cover all enhancements, regardless of artful packaging – the “whole work”
  - Circumvention should be technically cumbersome

opensource.com

GPL REQUIREMENTS

- Distribution of modified version must be under GPL
  - Exception for “mere aggregation”
- No imposition of “further restrictions” on downstream exercise of GPL rights
    - Corollary: liberty-or-death clause
- Accompany binaries with **complete corresponding source code** licensed under GPL
  - Amount of source code ≈ copyleft scope
- What a skilled developer needs to rebuild
    - System library exception

open source .com

GPL COPYLEFT SCOPE

- Interesting/difficult questions arise regarding GPL copyleft scope in various technical contexts involving combinations of components
- From a **legal risk** perspective, issues are mostly academic
- Projects and businesses should comply with GPL by making good faith effort to satisfy strong copyleft policy goals
- FSF continues to provide persuasive guidance; narrow interpretations are non-customary

opensource.com

WEAK COPYLEFT

- Originate in community criticism of strong copyleft
- Popular examples: LGPL, MPL and EPL families
  - LGPLv2.x second most popular FLOSS license
- Common features:
  - Copyleft scope (including source code requirement) limited to something less than GPL “whole work”
  - Can distribute proprietary executables
- Wide gap between LGPL text and liberal customary interpretation

PERMISSIVE (NON-COPYLEFT)

- Popular examples: BSD, MIT/X11, and Apache families
- Continuation of older public domain tradition + reaction against strong copyleft
- Policy goal: maximize downstream adoption, protect upstream developers from legal/reputational risk
- Derivative works licensable under more restrictive terms (proprietary, GPL if compatible)
- Notice requirements, but no source code requirement
  - But often strong social expectation to contribute some improvements upstream

open source .com

LICENSE ENFORCEMENT

- FLOSS licenses are generally assumed to be legally enforceable (cf. *Jacobsen v. Katzer*)
- Litigation risk is so low that compliance is motivated principally by ethical and social concerns
- Prior to 2000s, all license enforcement took place outside of court system
- Active GPL enforcement after 2000 focuses mainly on embedded device vendors, brought by small group of prominent licensors
  - Simple fact patterns: **no source code** – material violation

APPROACHING LICENSE COMPLIANCE

- Understand the **reasonable customary expectations** of upstream developers
- Downstream lawyers should avoid forcing community-developed licensing traditions into ill-fitting proprietary legal frameworks
- Downstream commercial users should become upstream contributors!
  - Developing good relationships with upstream communities minimizes enforcement risk and aids compliance
- Be wary of companies with “dual-license” business models

opensource.com

PRODUCT/PROJECT DEVELOPMENT

- FLOSS license compliance is usually easy once you figure out applicable license terms
- Both projects and vendors should exercise legal care in using third-party code, **as early as possible**
- Good software development practices lead to good license compliance
  - E.g. version control facilitates GPL compliance: you know exactly what sources were used to build a given binary
  - Developers should document how to generate build

open source .com

DUE DILIGENCE – INBOUND CODE

- Be able to reconstruct how code was put together and where it came from
- Biggest problem is device vendors obtaining firmware from suppliers without inquiry into licensing issues
- Transparency in use of third-party code aids diligence
  - Projects as well as commercial product developers benefit from explicit legal guidelines
  - Developers should not use prebuilt upstream binaries!

opensource.com

SOURCE CODE ANALYSIS

- Lawyers need to acquire some skills to extract legal information from source code
  - Understand how legal information is customarily recorded and presented by upstream developers
    - COPYING files, source code ‘headers’, GPL exceptions, disjunctive dual licensing, etc.
  - Identify external dependencies
  - Understand software build techniques
  - Determine who committed what

COMPLIANCE MECHANICS

- Notice requirements (esp. for permissive licenses)
  - For binary distributions, best practice is to maintain a text file that contains all required legal notices
- Source code requirements (copyleft licenses)

SOURCE CODE REQUIREMENTS: GPL

- 3-year written offer or accompany binary with source
  - Latter usually preferable for vendors except in embedded scenarios; always for projects
  - Don’t use offer to postpone dealing with problem!
  - FSF: for network distribution, can point to location hosted by third party (explicit in GPLv3)
  - Source offer not available for network distribution in GPLv3
- Must include build scripts and build instructions
- Should provide information on what compiler was used

SOURCE CODE REQUIREMENTS: OTHER

- LGPL: Can generally follow GPL rules; “suitable shared library mechanism” eliminates source requirement
- Other weak copyleft licenses: less detailed; assume written offer option not available
  - MPL-like licenses specify minimum post-binary-distribution time intervals

GPLv3 INSTALLATION INFORMATION

- Applies to binaries distributed in/for locked-down consumer products if the GPLv3 software is modifiable by a third party
- Vendor must provide information sufficient to allow skilled developer to install functioning modified versions on same device, with some limits
- No known enforcement experience
- Restoration of rights following **GPLv2** termination may be conditioned on providing such information too

opensource.com

GPL AND TERMINATION

- GPLv2 features automatic termination; key enforcement tool in US and Germany
- GPLv3 provides two explicit cure opportunities: permanent restoration of rights if:
  - no complaint 60 days after coming into compliance
  - cure within 30 days of receipt of notice of first-time violation
- May therefore be desirable to take “GPLv2-or-later” code as GPLv3 to take advantage of cure

opensource.com

rfontana@redhat.com