Make Your CCLA Easy To Use, Please!

Observations, rants, raves, and suggestions

Kevin P. Fleming, CTO Office

LF Open Source Leadership Summit 2017

February 4, 2017

techatbloomberg.com

Bloomberg

Introduction (who am I?)

- Multi-decade contributor to open source projects
- Project leader and community manager
- For the last 4.5 years, open source project coordinator at Bloomberg

techatbloomberg.com

Bloomberg

IANAL - IANYL - IDNPALOT

- I am not a lawyer.
- I am not your lawyer (see point above).
- I do not play a lawyer on TV.

techatbloomberg.com

Bloomberg

What's the easiest type of CCLA to use?

- The 'null' CCLA (aka the CCLA which does not exist)
- Failing that, the Developer's Certificate of Origin (DCO)

Are we done?

Unfortunately not, as many open source projects have been told that they need a CCLA.

Starting from the beginning...

What is the minimum number of people required to execute a CCLA at a large company?

That's right, four.

Yes, four.

Let's give them names:

- Alice: Alice is a developer who wants to contribute to the Frobnatz project
- Bob: Bob is her employer's general counsel (or equivalent)
- Claudia: Claudia is her employer's CTO, CSO, CIO, or some other C-level executive
- Kevin: Kevin is the guy who has to make all this work

Step 1: Alice requests permission to contribute to Frobnatz

(Yes, we all know that this is rarely the actual step 1, but let's live in fantasy land for a bit.)

Kevin gets the request, and hasn't heard of Frobnatz before. Time for some research.

What will Kevin need to be able to complete this task?

- A readable electronic copy of the terms of the CCLA
- Some way to have Bob (notice that Bob and Kevin are not the same person) review and approve the terms
- A way to have Claudia (notice that Claudia and Kevin are not the same person) sign the agreement once Bob has approved it
- A way to indicate to the Frobnatz project that Alice will be contributing under these terms, and some method they can use to identify her contributions

In reality, what has Kevin found?

- CCLA text only visible in Adobe Sign (formerly EchoSign) or DocuSign
- CCLA text only visible after logging in to a social network
- No CCLA, even though the project has an ICLA
- CCLA which requires the contributor list to be included as an addendum
- CCLA which grants more permissions to the project than to its users
- CCLA which covers tens (or hundreds) of projects at once
- Many more, most of which Kevin has conveniently forgotten

Bloomberg

What do we really need?

CCLA text

Available to the public, no need for any login or registration.

In a simple, free-to-use form (preferably ODT or plain text, but PDF is acceptable).

Includes at least a version number, if not something more comprehensive (such as an SHA-1 hash). A simple date, or only a year (shudder), is insufficient.

What do we really need?

CCLA terms

Does **not** consist of 'roll your own' terms. This will only serve to complicate and lengthen the execution process. An existing, well-understood CCLA should be used, as would be done when choosing an open source license. The Apache Software Foundation CCLA is a good starting point.

Must not require that contributors give the project more permissions than it gives to its users. The incoming terms and permissions should be identical to the outgoing terms and permissions. If they aren't, many companies will not permit their employees to contribute.

What do we really need?

Ease of execution

Ideally, executable via sign/scan/email. FAX is acceptable for projects still living in the 1990s.

Any online execution (signing) system must not require creation of an account, or linkage to any sort of third-party authentication provider.

Bloomberg

What do we really need?

Ease of execution (online execution)

If an online execution platform is used, it must allow Kevin (not Claudia or Bob) to populate the agreement and indicate acceptance by Claudia, but then send Claudia an email message allowing her to verify that the agreement should be executed.

The online execution platform should include the same version number (or hash) as seen on the readily-available copy of the text, and a completed (executed) copy of the CCLA should be sent to the submitter once it has been accepted by the project.

What do we really need?

Contributor lists

The 'permitted contributor' list should be updatable by Kevin, without requiring re-execution of the CCLA.

The list must include some form of identifier that is controlled by the employer (company email addresses work well for this), and the project should verify contributions against these identifiers.

It must be possible to **remove** contributors from the list.

What do we really need?

Contributor lists (in fantasy land)

If the project chooses to use email addresses as identifiers, and is operated by a medium-to-large organization which also hosts other projects, it would be incredibly friendly to allow companies (and others) to register their domain names so that contributions made by employees who have not been added to the CCLA can generate a notification to Kevin.

What do we really need?

Version updates

If for some reason the project needs to modify the text of its CCLA, that must cause a new version number to be generated.

In addition, all existing submitters of CCLAs should be notified that the new version exists, and be given a suitable 'diff' so that they can quickly and easily review the changes.

If such a change is being made by the project and will require that all existing contributors execute new CCLAs (rare, but it could happen), substantial advance notice should be provided to all previous CCLA submitters.

What do we really need?

Consistency

There are some large, well-known, but unnamed organizations that have blanket policies about the use of ICLAs and CCLAs for their member projects. These policies are clear, easy to understand, and logical.

Bloomberg

What do we really need?

Consistency

However, because the actual mechanics of executing CCLAs with these organizations can be cumbersome and time-consuming, some of their member projects have chosen to use 'relaxed' policies. For example, they only require a CCLA when a contributor graduates to having 'commit' privileges, as opposed to only being able to submit code via an issue tracker or other submission system. Interestingly, they require an ICLA from these same committers before they can submit through the issue tracker, even though the legal terms in the ICLA and the CCLA are nearly identical (and the differences are not related to having 'commit' privileges).

What do we really need?

An API!

For the programmers in the audience, what I'm asking for here is an API. An API that's easy to use correctly, and hard to use incorrectly, and which (as much as possible) follows the APIs of other projects, so that the user (me) doesn't have to learn your API from scratch.

Summary

We really do want to contribute to your projects: don't make it hard.

When an employee wants to contribute, if the process to make that possible will require hours of effort over days (or weeks, depending on executive schedules), the company will have to decide whether the potential contributions warrant that effort. We don't want to have to do this.

Just as project leaders, community managers, and others should always be thinking about how easy it is for new contributors to get their first patch into the project, they should be considering the possibility that that new contributor might be contributing on behalf of a large, complex, and very busy company, and that that company might be trying to foster a long relationship with the project (and many, many contributions).

Questions?

techatbloomberg.com

Bloomberg

To learn more about technology at Bloomberg, check out our blog

To learn more about engineering careers at Bloomberg, check out our Careers site

techatbloomberg.com

Bloomberg