NZGOAL-SE public consultation report

Open Government Information and Data Programme

30 March - 30 April 2016

Foreword

Public consultation was undertaken by the Open Government Information and Data Programme for the extension to NZGOAL for open source software licensing. This was carried out between the 30th March and 30th April. An open, transparent and participative approach was taken using the online discussion and consensus building tool, Loomio. The discussions remain available online as a record of what was discussed and how revised policy decisions were made.

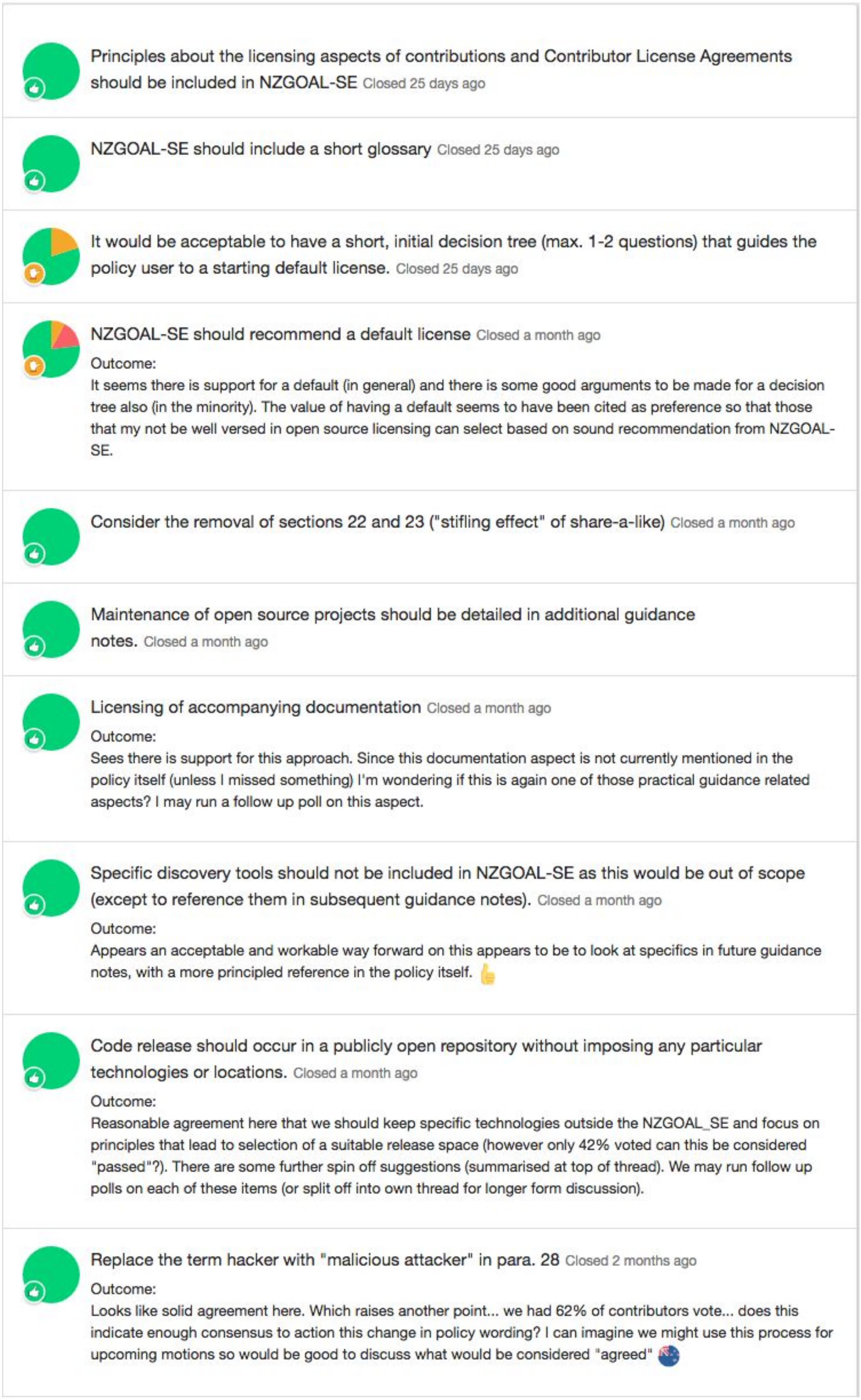
This can be accessed at the URL:

https://www.loomio.org/g/NohQxyr9/nzgoal-software-extension-discussion-of-draft

This following report includes:

1. A summary of the proposals voted on by participants and the outcomes.
2. Analytics report of the engagement that occurred during the public consultation period.
3. A verbatim archive of the conversations.

Proposal summary



Principles about the licensing aspects of contributions and Contributor License Agreements should be included in NZGOAL-SE Closed 25 days ago

NZGOAL-SE should include a short glossary Closed 25 days ago

It would be acceptable to have a short, initial decision tree (max. 1-2 questions) that guides the policy user to a starting default license. Closed 25 days ago

NZGOAL-SE should recommend a default license Closed a month ago

Outcome:
It seems there is support for a default (in general) and there is some good arguments to be made for a decision tree also (in the minority). The value of having a default seems to have been cited as preference so that those that my not be well versed in open source licensing can select based on sound recommendation from NZGOAL-SE.

Consider the removal of sections 22 and 23 ("stifling effect" of share-a-like) Closed a month ago

Maintenance of open source projects should be detailed in additional guidance notes. Closed a month ago

Licensing of accompanying documentation Closed a month ago

Outcome:
Sees there is support for this approach. Since this documentation aspect is not currently mentioned in the policy itself (unless I missed something) I'm wondering if this is again one of those practical guidance related aspects? I may run a follow up poll on this aspect.

Specific discovery tools should not be included in NZGOAL-SE as this would be out of scope (except to reference them in subsequent guidance notes). Closed a month ago

Outcome:
Appears an acceptable and workable way forward on this appears to be to look at specifics in future guidance notes, with a more principled reference in the policy itself. 👍

Code release should occur in a publicly open repository without imposing any particular technologies or locations. Closed a month ago

Outcome:
Reasonable agreement here that we should keep specific technologies outside the NZGOAL_SE and focus on principles that lead to selection of a suitable release space (however only 42% voted can this be considered "passed"?). There are some further spin off suggestions (summarised at top of thread). We may run follow up polls on each of these items (or split off into own thread for longer form discussion).

Replace the term hacker with "malicious attacker" in para. 28 Closed 2 months ago

Outcome:
Looks like solid agreement here. Which raises another point... we had 62% of contributors vote... does this indicate enough consensus to action this change in policy wording? I can imagine we might use this process for upcoming motions so would be good to discuss what would be considered "agreed" 🇳🇿

Consultation analytics

1. Summary statistics
2. Referring websites
3. Landing pages and visits
4. Participant interaction
5. Desktop/Mobile visits

Analytics report: NZGOAL-SE

30 March – 30 April 2016

Group members: 37

Discussions: 16

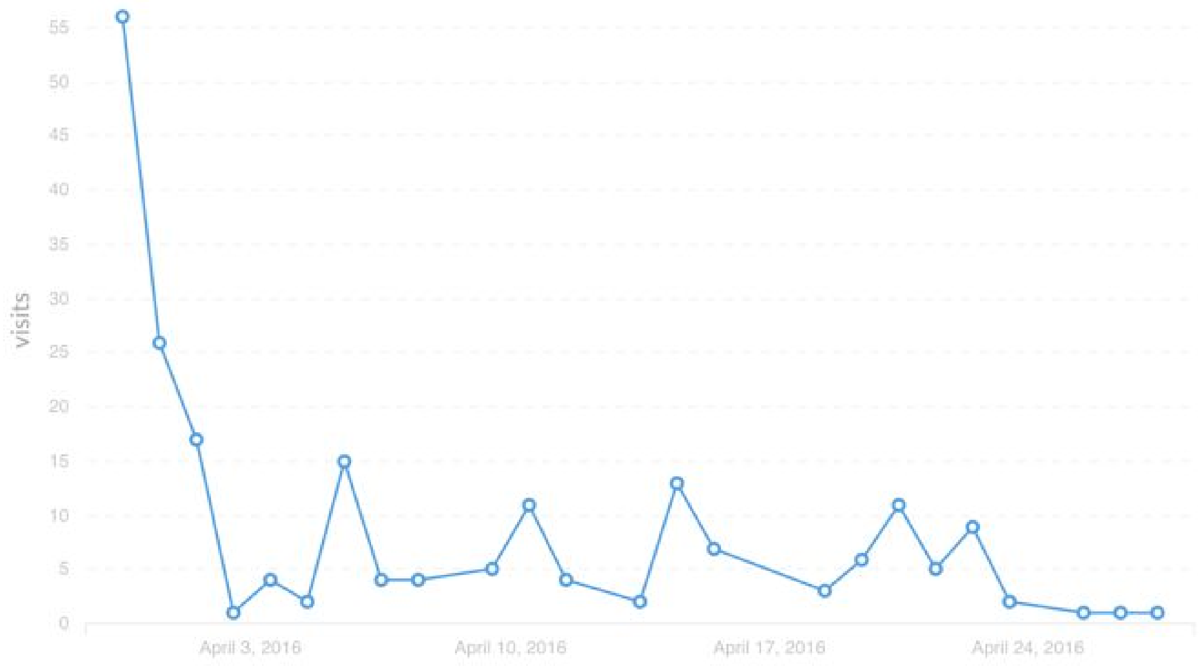
Comments: 175

Proposals: 10

Total visits: 353

Avg. time on page: 3:02

Total visits per day



Referring websites

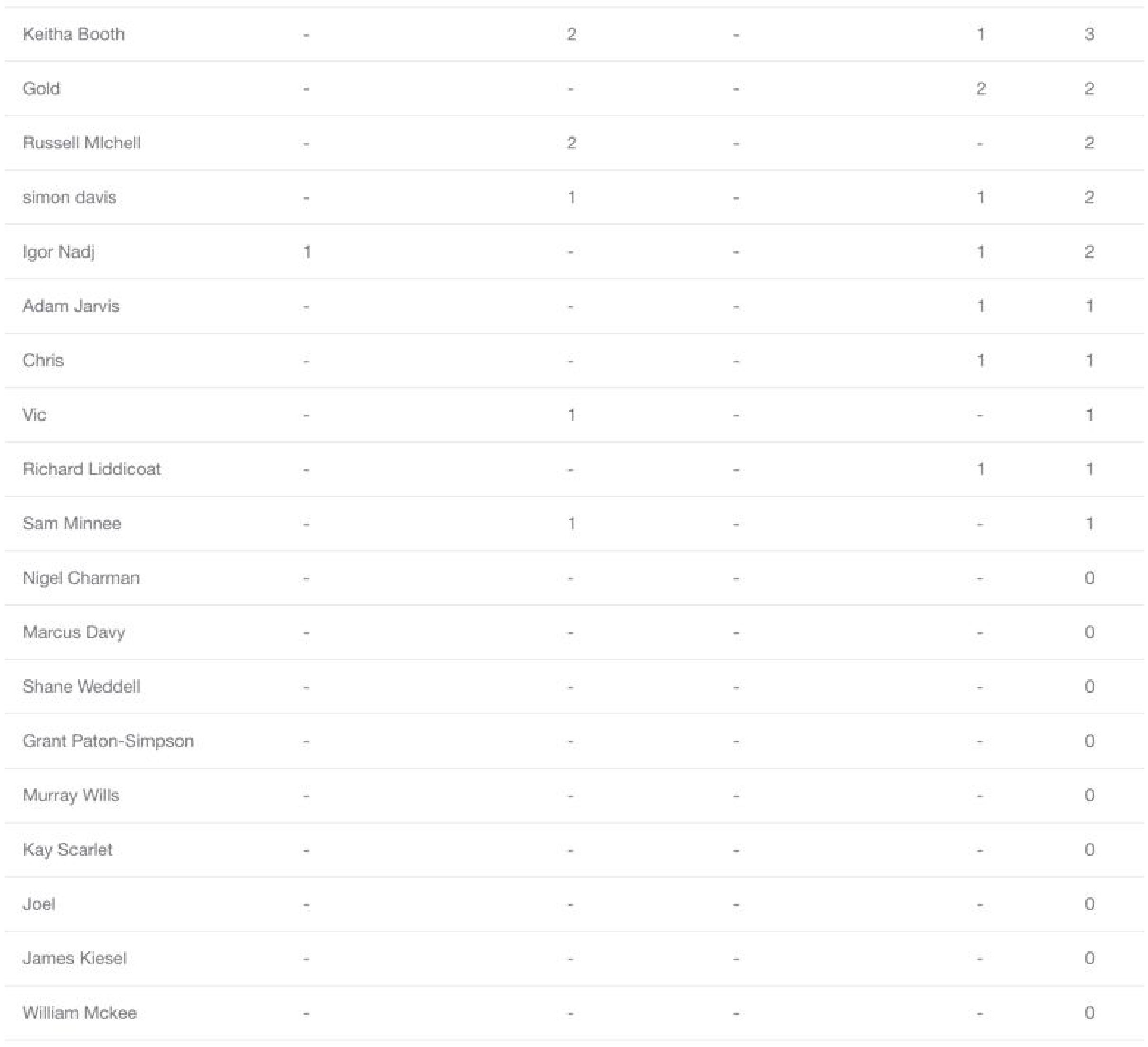
Referrer	Visits
https://www.ict.govt.nz/guidance-and-resources/open-government/new-zealand-government-open-access-and-licensing-nzgoal-framework/nzgoal-se/	10
http://groups.open.org.nz/groups/ninja-talk/messages/topic/37f1P9F34Jpi2z5eDgZ0xR	4
https://mail.google.com/mail/u/0/	4
http://groups.open.org.nz/groups/ninja-talk/messages/topic/37f1P9F34Jpi2z5eDgZ0xR/	1
etherpad.nzoss.org.nz/CouncilMeetingApr2016	1
github.com/	1

Landing page and number of visits

Landing page	Visits
https://www.loomio.org/g/NohQxyr9/nzgoal-software-extension-discussion-of-draft	111
https://www.loomio.org/g/NohQxyr9/nzgoal-software-extension	18
https://www.loomio.org/d/54Vxapcb/status-of-gpl-in-govt-software-projects	17
https://www.loomio.org/d/54Vxapcb/status-of-gpl-in-govt-software-projects?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_comment	14
https://www.loomio.org/d/3ogWN9I2/ongoing-maintenance-of-foss-projects?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_comment#comment-973261	12
https://www.loomio.org/dashboard	9
https://www.loomio.org/d/Qdgb7w25/section-1-purpose-discussion?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_discussion	9
https://www.loomio.org/d/54Vxapcb/status-of-gpl-in-govt-software-projects?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_comment#comment-978150	5
https://www.loomio.org/users/sign_in	4
https://www.loomio.org/d/IL4X72AY/source-code-project-catalogue-discovery-tool?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_discussion	3
https://www.loomio.org/d/54Vxapcb/status-of-gpl-in-govt-software-projects?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_comment#comment-998626	2
https://www.loomio.org/m/cBpjOaz2/votes/new?position=yes&utm_campaign=thread_mailer&utm_medium=email&utm_source=new_motion	1
https://www.loomio.org/d/GPt5e68k/when-would-you-pick-gpl?utm_campaign=thread_mailer&utm_medium=email&utm_source=new_comment#comment-986491	1
https://www.loomio.org/d/TfnmWEr5/alternative-version-control-repositories?proposal=jVpR7cJd	1
https://www.loomio.org/m/ZSKRuPFQ/votes/new?position=yes&utm_campaign=user_mailer&utm_medium=email&utm_source=missed_yesterday	1
https://www.loomio.org/d/3ogWN9I2/ongoing-maintenance-of-foss-projects?proposal=ikbznrH1	1
https://www.loomio.org/d/3ogWN9I2/ongoing-maintenance-of-foss-projects	1
https://www.loomio.org/inbox	1
https://www.loomio.org/d/SEshIfzW/use-of-the-term-hacker?utm_campaign=thread_mailer&utm_medium=email&utm_source=user_mentioned	1
https://www.loomio.org/d/SEshIfzW/use-of-the-term-hacker?proposal=ZSKRuPFQ	1

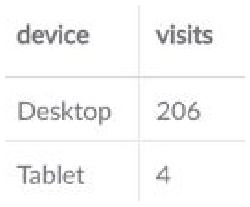
Number of interactions per participant

name	discussions started	comments	proposals started	votes	total interactions
Cam Findlay	6	49	9	9	73
Dave Lane	1	26	-	6	33
Strypey	-	21	-	7	28
Edward Abraham	2	11	-	4	17
Brent Wood	2	6	-	5	13
Don Christie	-	11	-	1	12
Rob Elshire	-	5	-	7	12
Tom Clark	1	6	-	3	10
Roy Storey	-	7	-	2	9
Cameron Shorter	1	6	-	1	8
GasparI	-	3	-	3	6
Jason Ryan	1	3	-	2	6
Sam Bonner	-	3	-	3	6
Byron Cochrane	-	2	-	3	5
Richard Best	-	5	-	-	5
T. Charles Yun	-	-	-	4	4
Finlay Thompson	-	1	-	3	4
Open Data NZ	-	3	-	-	3



Keitha Booth	-	2	-	1	3
Gold	-	-	-	2	2
Russell MIchell	-	2	-	-	2
simon davis	-	1	-	1	2
Igor Nadj	1	-	-	1	2
Adam Jarvis	-	-	-	1	1
Chris	-	-	-	1	1
Vic	-	1	-	-	1
Richard Liddicoat	-	-	-	1	1
Sam Minnee	-	1	-	-	1
Nigel Charman	-	-	-	-	0
Marcus Davy	-	-	-	-	0
Shane Weddell	-	-	-	-	0
Grant Paton-Simpson	-	-	-	-	0
Murray Wills	-	-	-	-	0
Kay Scarlet	-	-	-	-	0
Joel	-	-	-	-	0
James Kiesel	-	-	-	-	0
William Mckee	-	-	-	-	0

Desktop / Tablet / Mobile visits



device	visits
Desktop	206
Tablet	4

Conversation Archive

1. Status of GPL in Govt software projects
2. Ongoing maintenance of FOSS projects
3. Use of the term "hacker"
4. Alternative version control repositories
5. Add advice for developers section
6. Learning from overseas discussion
7. Source code/project catalogue (discovery tool)
8. When would you pick GPL?
9. When would you pick MIT?
10. Code forking (section 29-30)
11. Section 1 & 2 (Purpose) - discussion
12. Business Case for each recommendation
13. Legal template for contracting
14. Modify the ICT procurement policy
15. Consider removing section 22 & 23
16. OSS or FOSS?
17. Adaptions (para. 19) should cover using same license as adapted project.

Status of GPL in Govt software projects

Started by **Brent Wood** 2 months ago · Public

The original draft, in sections 21/22/23 discusses the relevance of the GPL for Govt open software. This thread is to discuss this content & propose alternatives.

PREVIOUS PROPOSALS

NZGOAL-SE should recommend a default license

Started by **Cam Findlay** · Closed 23 days ago

There has been some discussion in this thread about the current draft NZGOAL-SE default license (MIT) and alternative approaches.

Two key alternatives were 1) having the default as copyleft (GPL) and; 2) having both MIT & GPL as options with some attributes or decision tree to pick a starting default (then work through the rest of the release process).

This poll is to get a quick read on whether we feel it important that NZGOAL-SE should recommend a *default license* and provide guidance on when to pick the alternative.

I'm purposefully not putting forward a particular license (GPL or MIT), as we are looking to understand if having a default in general may be of practical value to public sector people following NZGOAL-SE. Please keep on topic 😀

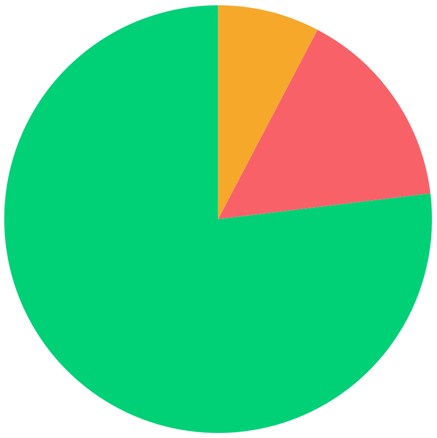
Based on the votes and discussion here I'll look to follow up with further dialogue and polls on the other alternatives 👍

Thanks.

OUTCOME

It seems there is support for a default (in general) and there is some good arguments to be made for a decision tree also (in the minority). The value of having a default seems to have been cited as preference so that those that my not be well versed in open source licensing can select based on sound recommendation from NZGOAL-SE.

POSITIONS



10 Agree

1 Abstain

2 Disagree

0 Block

35% of members have stated their position (13/37)

Edward Abraham agreed:
A clear (well reasoned) recommendation helps everyone. I don't want to have to discuss the ins and outs of MIT versus GPL more than necessary.

Finlay Thompson agreed:
Having a default position will make it a lot easy for smaller projects. I agree with Don though that the specific advice matters. However this proposal is that there should be a recommended license(s).

Dave Lane agreed:
I think it's important to give some guidance - I think most entities will use the default, as many won't know much about the options (based on a LOT of past experience).

Sam Bonner agreed.

Gasparl agreed:
This makes sense for ease and consistency as most people are not experts in license issues and why copyleft is preferable to serve the community. One alternative (e.g. LGPL) should be included for special cases (e.g. libraries).

Brent Wood agreed.

Rob Elshire agreed.

T. Charles Yun agreed:
i think a default is a good idea. it establishes more than guidance, it sets a position on the fact of assumption of an open license. stating GPL or MIT as option is a supportable. a separate doc could go into details on how/why to pick.

Cameron Shorter agreed:
The government policy should step up and tackle the hard question of which license to recommend so that time and effort isn't wasted debating and potentially introducing incompatible approaches.

Chris agreed:

Better to have an option so you can release with caveats, to promote sharing rather than may it an all or nothing sum.

Cam Findlay abstained.

Strypey disagreed:
For me it's horses for courses. Code license choice is to some extent political (as evidence by the reaction to the GPL critique sections). The best course is to clearly explain the rationale for (and consequences of) choosing each license in a set

Don Christie disagreed:
Without knowing what that recommendation would be I find it hard to support this idea. We have had some good clear discussion on the two approaches that could be stated quite clearly.

Collapse

ACTIVITY

Cam Findlay in reply to **Edward Abraham**

This could make a good additional to the thread on "when you would pick GPL", would you mind cross-posting there @edwardabraham ?

Edward Abraham in reply to **Cam Findlay**

I will see if I can find out more for you about the current status of the discussiins around this ...

Cam Findlay in reply to **Edward Abraham**

Thanks, Only as long as it's public knowledge @edwardabraham and not anything in confidence please. Keep in mind the purpose of the consultation process we're working through together here 😀

On another note, when you mention "*other developments get contributed back to the project*" in your story about the mapping code, keep in mind that the share-a-like clause of GPL only comes into play should a modified copy of the code be *redistributed*. It doesn't enforce contributions back to the original project per say, which is a common misconception of GPL.

Edward Abraham likes this.

Cameron Shorter agreed:
The government policy should step up and tackle the hard question of which license to recommend so that time and effort isn't wasted debating and potentially introducing incompatible approaches.

a month ago

DC **Don Christie**

Given the consensus and experience in this discussion seems to favour GPL does that mean dropping MIT as the recommended licence?

Don Christie disagreed:
Without knowing what that recommendation would be I find it hard to support this idea. We have had some good clear discussion on the two approaches that could be stated quite clearly.

a month ago

Finlay Thompson agreed:
Having a default position will make it a lot easy for smaller projects. I agree with Don though that the specific advice matters. However this proposal is that there should be a recommended license(s).

a month ago

RE **Rob Elshire**

Having a default license makes good sense to me. As others have said, it makes things easier in (especially) smaller projects. Regardless of which license is the default, the SE should give clear reasoning behind choosing a license.

Rob Elshire agreed a month ago

a month ago

T. Charles Yun agreed:
i think a default is a good idea. it establishes more than guidance, it sets a position on the fact of assumption of an open license. stating GPL or MIT as option is a supportable. a separate doc could go into details on how/why to pick.

a month ago

DC **Don Christie** in reply to **Rob Elshire**

Save me from "easy", please. It usually doesn't mean "right". As our mothers surely told us all.

So, let's get this right and clear and avoid easy for easy sake.

RE **Rob Elshire** in reply to **Don Christie**

Point taken. What, in your view, would right and clear in this context look like?

Strypey disagreed:
For me it's horses for courses. Code license choice is to some extent political (as evidence by the reaction to the GPL critique sections). The best course is to clearly explain the rationale for (and consequences of) choosing each license in a set

a month ago

Strypey in reply to **Richard Best**

Thanks @richardbest for your detailed feedback. It's actually quite refreshing to be in a discussion of free code licensing and feel like the under-informed one for a change 😀 I hope you don't mind if I reply to your questions a little out of order.

Status of GPL

Like others, I agree that the paragraphs 22-23 can go.

Selection of the MIT licence as the non-copyleft licence

Just out of curiosity, when you say:

> what most people consider to be the "MIT licence"

Which license is this? The X11 or the Expat? It's good to know the FSF's caution was taken into account. If you are confident that NZ GOAL-SE encouraging people to call it the "MIT License" will not create confusion, I can live with it.

AGPL

On the subject of AGPL, yes, I was thinking of the GNU AGPLv3 you linked to, which incidentally is the license Loomio is released under. I recommended this license to the Loomio developers to protect them from the risk of Company X setting up a "software-as-a-service", using a version of the Loomio software they modified to add new features, without sharing their code back to the Loomio team. My understanding was and is that because Loomio runs on a server, the copyleft provision of GPLv3 would not be triggered to allow the Loomio developers to benefit from the third party modifications, unless Company X distributed pre-compiled binaries (or possibly source code) of the modified version. In short, "software-as-a-service" allows re-users to avoid honouring the spirit of the GPL. My understanding is that AGPL fixes this bug by specifying that to allow users access to a running version of the software on Company X's server counts as "distribution" triggering copyleft.

If my take on this is correct (and IANAL so it needs checking), then NZ GOAL-SE needs to advise that if developers are releasing "server-side software" (software running on a remote computer and accessed via a network) *and* they want to be legally guaranteed access to (and re-use of) any modified version of their software offered by third parties as a service, they need to choose AGPL. Otherwise they might as well choose a non-copyleft license.

Patent-blocking abilities of Apache 2.0

On the subject of Apache 2.0, I acknowledge that the changes to the NZ Patent Act mean nobody will be sued for patent infringement in NZ Courts. But open source projects routinely span multiple jurisdictions, many of which do still enforce software patents. My understanding (again IANAL) is that Apache 2.0 protects re-users of the software not only from patent infringement suits originating from the original developer (in cases relevant to NZ GOAL-SE a government agency) but from any third party distributing modified versions. It does this by including a provision that revokes the right to use the original code under the license term, which is triggered by the filing of patent infringement litigation (see here: http://en.swpat.org/wiki/Patent_clauses_in_software_licences#Apache_License_2.0). Like the GPL, it's more complicated because it does a more complicated job.

Again, assuming I haven't got my wires crossed here, imagine a local library in the US starts using (and contributing back to) a piece of software licensed non-copyleft by a public library in NZ. Company Y starts vending a modified version of the software, and files a software patent on some parts of what that software does with the US Patent Office. Company Y then sues the local library in the US for patent violation. Most non-copyleft licenses don't address this, so if the software is licensed under any of those, the library is forced to either a) abandon the software and find an alternative, b) attempt to strike down the patents, or c) start paying Company for their version of the software, all of which would cost money and cause stress. My understanding is that if the software were licensed under Apache 2.0, Company Y would lose all rights to use the original code as soon as they filed, which would make their modifications useless, and most likely invalidate their patents. Company Y would either have to a) write their own software, b) use software under a more "permissive" non-copyleft license to enable their patent trolling or c) stick with the Apache 2.0 licensed software but use a less predatory business model.

LGPL

Brent brings up the question of using LGPL for libraries. Just to clarify, libraries here are defined as pieces of code providing a discrete function or set of functions, which can be called on by other programs (eg code to turn a song into an .MP3 file), not building that lend books 😉
TL;DR of the FSF position on the pros and cons of using the LGPL seems to be that using LGPL for a free code library may make it more likely to be used instead of proprietary libraries offering the same functions (and popularity has many benefits), while if a free code library that offers significant new functionality uses GPL, this has resulted in other software choosing to use GPL, increasing the amount of free code available, and its re-use.

Preference for non-copyright licence as default over GPL unless GPL-licensing required or there is a compelling reason to use the GPL

I think the third approach you mention is the way to go. If any one free code license could cover all use cases, I think the vast majority of projects would be using it by now, yet a number of different licenses continue to be popular (as is non-licensing but that's another story). My suggestions would be:

- Explain the practical differences between copyleft and non-copyleft
- GPLv3 as the recommended copyleft license for "client-side applications" (software running on the users computer, whether desktop, laptop, mobile, or otherwise)
- AGPLv3 as the recommended copyleft license for "server-side applications (as defined above)
- Explain the differences between non-copyleft licenses that provide protections against patent threads and those that don't
- Apache 2.0 as the recommended non-copyleft license for protecting against patent threats in other jurisdictions (for reasons given above). If the NZ approach to software patents spreads around the world (we live in hope), then this could be dropped from future revisions. On the other hand, if implementing the TPP results in the NZ government restoring general software patents in NZ, future revisions could recommend Apache 2.0 as the only non-copyleft license
- "MIT License" as the recommended non-copyleft license where patent threats in other jurisdictions isn't considered a significant risk (just to be clear I don't have a strong preference for MIT but your reasoning for choosing it over any of the BSD licenses seems sound)
- Explain the practical differences between licensing a library under GPL (full copyleft), LGPL ("weak copyleft"), or Apache 2.0 (non-copyleft)

Cam Findlay likes this.

Cam Findlay in reply to **Strypey**

Well thought out comment @strypey 👍 - I'll take some time to digest this.

If you're up for it, I'd be really interested to see what a decision tree based on your suggestions might look like as a first step of the current draft tree on page 19 of the draft. Is this something you might be able to put together (even in a rough form) and share with us here? 😀

Cameron Shorter

Re license selection, the Australian government licensing recommendations for Open Data includes a recommendation on software licensing. I suggest it would be wise to align NZ recommendations with those of other Governments to facilitate international compatibility of policies.
http://www.ausgoal.gov.au/BSD-and-Other-Software-Licences

Quoting:
"If your organisation is publishing software developed without the inclusion of software from any external source, AusGOAL recommends the BSD 3-Clause Software Licence. It provides permissions akin to the CC Attribution Licence, and being one of the Open Software Foundation recommended licences, is well recognised within the open software community. Software licensed under the BSD Licence can be incorporated onto other open source projects, including those licensed under the GPL and Apache Licences. However, GPL licensed software cannot be incorporated into software licensed under the BSD Licence."
"If your organisation is publishing software that incorporates elements from other open source software projects, such as a project licensed under a GPL Licence, AusGOAL recommends application of the same licence under which you obtained third party code."

Cameron Shorter

PS, I suggest also reaching out the the author behind Australian Open Licensing, Baden Appleyard,
http://www.ausgoal.gov.au/contact-us

Cam Findlay and Strypey like this.

KB **Keitha Booth**

NZGOAL built on initial AGILF work and AUSGOAL picked up much of NZGOAL. So each has taken from the other. It may be now our turn to take the lead with respect to software licensing. Certainly consider AUSGOAL recommendations seriously but independently.

Strypey likes this.

DC **Don Christie** in reply to **Cameron Shorter**

Hi

I understand the logic but I believe it is flawed. As we have discussed elsewhere the original licence of an open source work is important. It seems odd to have a document recommending open sourcing software only to do so in a manner that allows it to be closed off immediately.

I think Australia have got this advice wrong...just as they did with their approach to the Common Web Platform. NZ did a better job there (not just because they chose Silverstripe :-)) but because they kept supplier participation very open and so acceptance has been wider deeper and more interesting than in Aussie.

So goes for recommending a licence. If you want to go down that route then in this case it really has to be a GPL approach. Otherwise you risk minimising the profound impact this policy and advice could have in government shared use of IT in NZ.

Cheers
Don

Dave Lane likes this.

Strypey in reply to **Cam Findlay**

I have had a go at a decision tree:

1) non-copyleft or copyleft?

non-copyleft: you simply want to make the code available, for any purpose, under very few conditions

- 2) Do you want to take steps to prevent your source code being used as part of the basis for a software patent litigation? (all copyleft licenses recommended below take as strong a stand as they can against software patents)
  - yes: Apache 2.0
  - not worried: "MIT License" (X11 or Expat) or "BSD License" (Modified or FreeBSD)

copyleft: you want to create a commons that others will draw from and must offer improvements back to if they publish their changes

- 3) Is the software you wish to release under a copyleft license a:
  - client-side application (programs that run on a desktop, laptop, or mobile device): GNU GPL
  - server-side application (programs that run on a remote server, accessed via a network): GNU AGPL
  - software library (code implementing a function or set of functions usable in other programs)
    - there are few or no other libraries that implement the same functions and you'd like programs to use a copyleft license if they use your library: GNU GPL
    - there are proprietary libraries that implement the same functions, so you don't mind your library being used in proprietary programs, so long as distributed modifications to the library itself are released as copyleft: GNU LGPL

I have also put this in table form in my wiki page of advice for projects releasing code as free software.

Strypey

Just to throw another cat among the Pidgeons, today I stumbled upon another license for server-side software. The Common Public Attribution License](https://www.socialtext.net/open/cpal_faq) is similar to the GNU AGPL, in that in both cases the deployment of the software as a network service triggers the copyleft provision (but with slightly different mechanisms). The difference is that the CPAL also obliges organisations using CPAL-licensed software as a network service to put some kind of "powered by [insert software name here]" badge on their user interface.

That could add another branch to the decision tree, under server-side software, asking whether the code releaser want the "powered by" badge when their software is deployed (CAPL) or don't case (AGPL).

Cameron Shorter

I agree with Keitha Booth, certainly consider AUSGOAL recommendations seriously but independently. I am Australian and hope to see this New Zealand document "get it right" and for Australia to follow suite.

Apache has been discussed as a possible license selection. I strongly encourage against Apache, as "it does not require you to include the source of the software". http://www.apache.org/foundation/license-faq.html#WhatDoesItMEAN I.e. Government could contract a company to write Widget X under an Apache license, and the company could deliver the widget as binaries without providing source, which is contrary to the Government's best interest.

Don Cristie, while I like the GPL license and personally would be impressed to see governments recommending it as a default position, the business logic which resulting in NZGOAL recommending CC-By for Data would likely be applicable for Software, resulting in the selection of a permissive license such as BSD. A GPL license is very prescriptive of the business models it imposes upon the NZ Software Industry. I've looked quickly, but can't seem to locate NZGOAL's justification for selecting CC-By for data. I suggest it would be useful to reference it.

Srypey, regarding your proposed decision tree, I think it is missing an statement on what NZ Government considers to be in the best interest of NZ Government, backed by solid reasoning. We here are Open Source advocates, and even we are divided in our opinion on what license to select. This Open Source policy document should be making it easier for government officials to select a license.

I suggest decision tree should be: If <building on existing open source> follow license convention of the open source project. If <creating new material> select <X license> based upon <similar logic used to guide selection of CC-By for Data>.

DC **Don Christie** in reply to **Cameron Shorter**

Hi Cameron

In your reply to me you are already making assumptions about who will maximise the use of the software and why it might be released in the first place. I am keen to avoid that hence my opposition to a single licence recommendation.

You said - "A GPL license is very prescriptive of the business models it imposes upon the NZ Software Industry"...

This simply isn't the case. There are a myriad of ways businesses have been built around GPL based software. But what your approach risks doing is minimising the value *government* gets from sharing its software. That is a concern to me as a tax payer as well.

Cheers
don

Dave Lane and Strypey like this.

Cam Findlay

For those that have so far responded in the *affirmative* in the poll for the consideration of a *default licence* (regardless of copyleft/permissive angles), would it be acceptable to you to have instead a quick qualifying decision tree (max 1-2 questions) that brings the user/developer releasing the software to an initial default? They would then work through the rest of the decision tree from there already proposed in the draft (some alterations might be required once we review implications of this kind of change). *I may run this as a follow up poll later this week after some dialogue.*

My thoughts here are that:

I agree there is value in clear guidance that a default might give the user of the policy and I think a short decision tree could perhaps offer that same value (that is, brining the policy user that my not know the ins and outs of copyright law and open source licensing to a starting default *quickly*). I'd like to see these questions in plain English and watch out for technical jargon.

This would maintain the spirit of the wider NZGOAL framework, that of giving guidance for the release of copyright materials and when to use attribution and share-a-like aspects of licensing.

For NZGOAL proper (the policy for content and data not software that NZGOAL-SE extends), it seems that 2 qualifying questions are asked before putting forward CC-BY as the default (granted these are based around whether the agency has the copyright to be able to release the work). However keep in mind NZGOAL proper does set out a principle of "CC-BY as default".

As a user and developer of FOSS, I'm a fan of both MIT, GPL and a host of other licenses. However am in a privileged position of having some understanding of copyright law and FOSS licenses from experience and use. Others may not have this and that should not exclude them from making practical use of NZGOAL-SE.

It would be nice to find common ground here to help move the policy toward a place where it get the best outcomes for government sharing code and for the wider public reuse (both citizens and commercial reuse). 😀

Also just like to say, I've been impressed with the conversations over the past few weeks and there are some great constructive points and things to consider heading toward revisions of NZGOAL-SE. Keep them coming. 🇳🇿

Cam Findlay abstained 24 days ago

24 days ago

Dave Lane

I'd like us to select a copyleft license (GPLv3/AGPL) as the default license, but deferring to upstream licenses for derivative projects. The purpose being that the interest of the citizens/taxpayers are prioritised.

Strypey likes this.

Cam Findlay updated the proposal to close at 5pm Tuesday, 26th April 2016

24 days ago

Cam Findlay

Updated poll given that 25th April is a public holiday in NZ.

Gasparl agreed:
This makes sense for ease and consistency as most people are not experts in license issues and why copyleft is preferable to serve the community. One alternative (e.g. LGPL) should be included for special cases (e.g. libraries).

24 days ago

Strypey in reply to **Cameron Shorter**

> I strongly encourage against Apache, as "it does not require you to include the source of the software".

None of the non-copyleft licenses require sharing of source code, certainly not any of the licenses commonly referred to as the "MIT License" or "BSD License". This is pretty much the essential distinction between copyleft and non-copyleft licenses! This is the kind of difference I think NZ GOAL-SE should help to clarify, rather than being overly prescriptive about license choice.

> the company could deliver the widget as binaries without providing source

While I would prefer to see public agencies using copyleft licenses, with a non-copyleft license, the requirement that source code be delivered to the contracting agency (and/or a public-facing repository as discussed elsewhere) could (and by default should) be written into the contract. Also, as the Koha/ LibLime debacle illustrates, control of any trademarks relating to the software name and logos by the contracting agency (or another vendor-neutral stewardship body) should also be written into the contract.

> We here are Open Source advocates, and even we are divided in our opinion on what license to select.

This is why it seems unlikely that we can arrive at one default license to recommend for all use cases. There are good reasons that each of the different types of license are used, and this is what I've tried to illustrate in my decision tree. Keep in mind I supplied this tree in response to a request from @camfindlay1 , I'm not proposing it as a replacement for the entire draft of NZ GOAL-SE 😉

Dave Lane likes this.

Ongoing maintenance of FOSS projects

Started by **Tom Clark** 2 months ago · Public

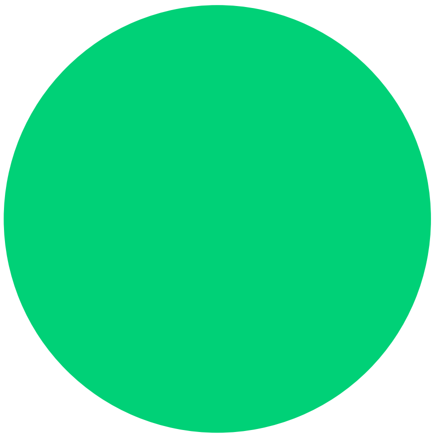
PREVIOUS PROPOSALS

Maintenance of open source projects should be detailed in additional guidance notes.

Started by **Cam Findlay** · Closed a month ago

Agreement here means that: you think that NZGOAL-SE should consider referencing a set of guidance notes around long term open source project maintenance, while keeping the detailed guidance itself outside of the policy itself. Guidance notes can more easily be updated to keep up with the times where as policy takes time to adjust and is best to contain the principles that lead towards good decision making.

POSITIONS



6 Agree

0 Abstain

0 Disagree

0 Block

17% of members have stated their position (6/35)

Strypey agreed:
As I said in comment, maintaining an open source project is a book-length topic, but perhaps a few general principles of good stewardship could be included, particularly as they relate to license choice.

Edward Abraham agreed.

Cam Findlay agreed.

Rob Elshire agreed.

T. Charles Yun agreed.

Richard Liddicoat agreed.

Collapse

ACTIVITY

Tom Clark

The document as drafted really only addresses the process to release software as open source. Perhaps that is it's intended scope. But the full lifecycle of an open source project includes ongoing maintenance of the project, including tracking issues, accepting contributions of code and other assistance, and communicating with the larger community that may coalesce around the project. It's important to advise organisations that plan to release software as open source to consider and plan for these issues.

Open Data NZ

A good point Tom. The focus has been on initial release, but I can clearly see that there is a need to plan for the full life cycle. I guess the question is do we incorporate that in this initial work, or do we flag it as a phase 2 project?

BW **Brent Wood**

If you want phase 1 to be relevant to phase 2, you need to deliberately do this. Short term solutions that don't think about tomorrow can cause more problem than they solve, requiring a baby AND bathwater fix for phase 2.

If NZGOAL-SE is intended at some time to take the long term success of a software project released as open source as one of its goals (I suggest it should), then it should factor this in from day 1.

In the GPL discussion I raised some issues about the requirements for ongoing success - releasing a software project which promptly goes belly up is a waste of everyone's time (& not just open source - Lotus 123 or Wordperfect anyone?)

Also - NZGOAL & the Declaration generally are about maximising value through reuse- any software project should be planned for the long haul if reuse is the goal. Software or data.

Tom Clark

When planning and performing the initial release, some thought must be given to the longer term plans for maintenance, updates, and community involvement. This document must at least bring up the topic.

As far as the question about how to run an ongoing FOSS project goes, that's a huge and complex topic. It probably warrants a document of it's own. For some organisations, it may be true that all they can manage to post their project code to GitHub and leave it there. They may not have the resources to manage a full fledged ongoing open source project. I think that that is ok, provided that the organisation is honest about the situation (honest with itself and with outside parties) from the beginning.

Strypey likes this.

Dave Lane

One extremely valuable outcome of this process would be for Gov't procurement and grant funding to recognise the need for maintenance/refactoring/infrastructure budget for the lifetime of the system in addition to any initial bulk outlay/funding.
At present, we're squandering many many resources and data sets created, for example, by our scientific community, because funding does not extend to them keep their results or other systems (e.g websites) running online indefinitely. It's a real mismatch.

Cam Findlay

FOSS community management and social aspects of shared maintenance could be better served in it's own set of guidance or borrowing from something like the online engagement guidance released on webtoolkit last year that makes mention of open source (reuse in action 😀)

See https://webtoolkit.govt.nz/guidance/online-engagement/

If we stick with just the licensing aspect of NZGOAL, I think what might be interesting to include is how to correctly accept contributions into Govt released projects and the licensing complexities around this. This addresses the shared maintenance aspects to some degree and leaves open for expansion in later revisions.

If a member of the public contributes some code to an existing nz govt code repository (though a Pull Request or some other peer review process), where does the copyright reside? Further, at what point is the code considered "licensed" from the member of the public to the original code repository maintainer? I believe copyright assignment in the NZ Copyright Act needs to be explicitly in "writing" (interpretation of that could be interesting in a digital world). So some sort of cross-licensing is going to be the primary mechanism here. Someone correct me if I'm wrong.

Having a way for contributions to be legally accepted and sub-licensed correctly to grow the works that is 1) not overly complicated and; 2) covers the legal aspects, would be worth addressing in the NZGOAL-SE.

I'd hope this would encourage contribution back to canonical code repositories and build communities around projects rather than further fragmentation and duplication (forking). 🇳🇿

Dave Lane

I think there has to be an element of responsibility for FOSS projects which refers back to the original service provider/vendor or someone else who has responsibility for the code base... I think there should either be a) an ongoing modest funding stream to ensure the service provider continues supporting the development of the project, b) an expectation that doing so is a "cost of business", or c) that projects gets handed over to an independent (funded) entity for maintenance after the initial procurement terms have been satisfied... I don't think copyright assignment is a good idea in a general case, as I think it inhibits contribution (mostly because it adds complexity).

Cam Findlay

@davelane on your last point here... agree, assignment is a pain.

This raises, how can FOSS code repos effectively communicate to contributors how the project can accept (legally) code contributions. From experience, I've often used a "contributing" file in my codebase that lets contributors know that by me accepting their code they are effectively licensing it to me and then I'm sub-licensing it back under the original license type of the projects canonical repo. I think this bit is unclear in the policy as it stands and think this is in scope due to the licensing aspect of it (the point of NZGOAL is licensing guidance after all).

This is because NZ Copyright Act has a first owner clause and no mechanism to waive copyright (also the reason we might not be able to utilise CC0 under the current legislation, that is a whole other discussion! 😀).

On the point about project maintenance, funding to maintain is one way, building a user community to share the maintenance is another ⭐

Cam Findlay started a proposal: Maintenance of open source projects should be detailed in additional guidance notes. a month ago

Edward Abraham

While a licence is relatively straight forward, the communities around a project are likely to be as diverse as the projects. I wouldnt want people to think that releasing code as open source requires them to run a community project. In many cases, it may simply be that the agency wants the code open sourced so the cant be held to ransom, without any expectation of it turning into an active project.

Cam Findlay likes this.

Edward Abraham agreed a month ago

a month ago

Strypey in reply to **Cam Findlay**

This is actually another advantage of using GPL or AGPL (EDIT: I've moved my off-topic rant on GPL moved to the relevant thread.

Cam Findlay agreed a month ago

a month ago

Rob Elshire agreed a month ago

a month ago

T. Charles Yun agreed a month ago

a month ago

Strypey

In response to the original topic, making open source projects work long term is more art than science, and a number of books have been written on the topic, including Producing Open Source Software by Karl Fogel, which you can download gratis from the website. There's no way this topic could be covered in any depth in NZ GOAL, but perhaps we could distil a few principles from sources like Fogel's book, to guide the way publicly-funded open source projects might be run?

My short list would be (*not* in order of importance):

- public-facing version repository: accessible to any potential contributor without any requirement to use proprietary software (we've covered this elsewhere)
- develop in public ("release early, release often"): no software project is every "finished" or "tidy", at least not for long, and part of the benefit of open source practice is avoiding developers duplicating each others' work, so it helps to see the code as it evolves. I've noticed that solo developers are particularly nervous about this, fearing their code will torn apart by scornful trolls, and sadly this can happen, which leads on to...
- have community standards with teeth: pre-defining procedures for decision-making, conflict resolution, and standards of behaviour is key. The model Code of Conduct put together by the Discourse project is a good template. A culture of friendly, welcoming, constructive, communication doesn't just happen, it needs to be envisioned and modelled by project leaders (both formal and informal), and violators put in the stocks and pelted with tomatoes (they knew that would happen, as it was laid out as the sanction for their action in the CofC). Which leads on to...
- community-driven (democratic? consensus?) decision-making: GIT helps with this because a fork can become the main project if the original maintainers are not responsive to the developer community, but this is not a surefire guarantee, as we've seen with the recent ragequit of a core BitCoin developer.
- consider the ongoing costs (in $ and time) of maintaining the project's toolset; repository, homepage, email lists etc and have a business model in mind. Even if that model is depending on grants, donations, or crowdfunding, if you know in advance that's likely to be the case, you can plan to keep your project infrastructure light.

Cam Findlay likes this.

Strypey agreed:
As I said in comment, maintaining an open source project is a book-length topic, but perhaps a few general principles of good stewardship could be included, particularly as they relate to license choice.

a month ago

Richard Liddicoat agreed a month ago

a month ago

The proposal has closed: Maintenance of open source projects should be detailed in additional guidance notes. a month ago

Cameron Shorter

Further: Guidance should be given to agencies as how to assess the level of open source engagement that should be invested in. Ie How much time/effort should a project contribute toward Open Source, and how should you justify the business case to put toward this investment. The answer will be different depending on the importance of the open source technologies to business goals. Investment is a sliding scale, covering: "report bugs", "provide patches", "release code", "participate in community", "lead community", "sponsor core development", "build community - through marketing, release management, answer external developer questions".

Strypey

I have a document on my wiki written some time ago, which describes some of the considerations for releasing code in a bit more details. It now includes the decision tree for choosing a license I drafted for another thread.

Use of the term "hacker"

Started by **Cam Findlay** 2 months ago · Public

Given there is a general move to making this term less derogatory given initiatives like GovHack and other such Hack-a-thon type events in govt, I move that the language here be changed. Either making it clear we are talking about "black hat" or using something along the lines of "malicious person".

PREVIOUS PROPOSALS

Replace the term hacker with "malicious attacker" in para. 28

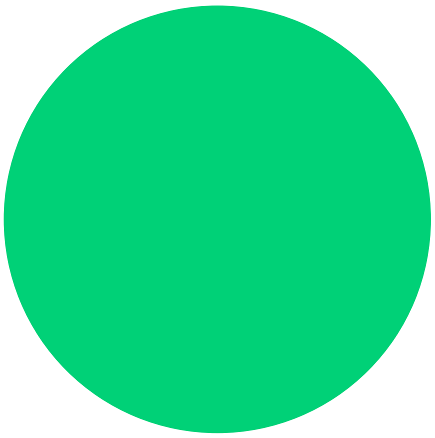
Started by **Cam Findlay** · Closed a month ago

The term is too casual and the meaning of the term hacker culturally has changed.

OUTCOME

Looks like solid agreement here. Which raises another point... we had 62% of contributors vote... does this indicate enough consensus to action this change in policy wording? I can imagine we might use this process for upcoming motions so would be good to discuss what would be considered "agreed"

POSITIONS



13 Agree

0 Abstain

0 Disagree

0 Block

62% of members have stated their position (13/21)

Gold agreed.

Tom Clark agreed.

Dave Lane agreed.

Cam Findlay agreed.

Sam Bonner agreed.

simon davis agreed: The term hacker is incorrectly used.
The correct term is cracker or adversary.
See https://tools.ietf.org/html/rfc4949

Byron Cochrane agreed.

Brent Wood agreed: No brainer - with govhack such a positive event

Jason Ryan agreed.

Igor Nadj agreed.

Rob Elshire agreed.

Roy Storey agreed: reduces ambiguity

Adam Jarvis agreed.

Collapse

ACTIVITY

Byron Cochrane agreed a month ago

a month ago

Brent Wood agreed: No brainer - with govhack such a positive event

a month ago

Adam Jarvis agreed a month ago

a month ago

Gold agreed a month ago

a month ago

The proposal has closed: Replace the term hacker with "malicious attacker" in para.

28 a month ago

Cam Findlay published a proposal outcome:

Looks like solid agreement here. Which raises another point... we had 62% of contributors vote... does this indicate enough consensus to action this change in policy wording? I can imagine we might use this process for upcoming motions so would be good to discuss what would be considered "agreed"

a month ago

Cam Findlay

I've raised a revision over at https://github.com/opendatanz/nzgoal-se/pull/2

KS **Kay Scarlet**

agree with wordage around hacker v malicious attacker. Minor point, it's usually hackathon not hack-a-thon 🙂

Alternative version control repositories

Started by **Jason Ryan** 2 months ago · Edited · Public

Propose that options in addition to Github are mentioned....

Principles guiding the choice of code repository for publicly-funded software:

- repository provides full public access to source code, to promote discovery and re-use
- respository software support government standards for accessibility for a full range of users
- repository software is itself free code, with no proprietary dependencies, so a user can access it without being required to use any proprietary software, either directly or indirectly
- repository software implements all relevant open standards
- respository may be self-hosted by the project or department developing it, or another public organisation, or a gratis public service hosted by an external provider
- projects and the repository they are using should be listed on a regularly-updated list of free code projects used across all-of-government, to increase standardization and reduce unnecessary duplication of effort

PREVIOUS PROPOSALS

Code release should occur in a publicly open repository without imposing any particular technologies or locations.

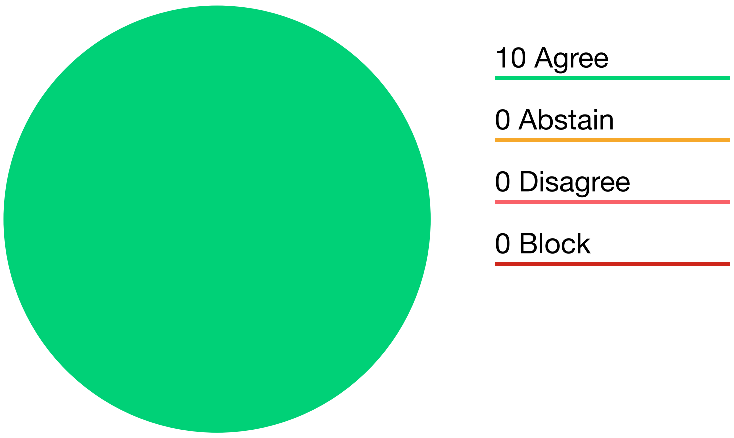
Started by **Cam Findlay** · Closed a month ago

Agreement means that you'd like to see the policy wording changed to indicate that the desired behaviour here is release of code in an open and public digital space. This does not include what technology platform to use, nor where this is located online (this would be up to the agency to decide though there might be guidance given outside the NZGOAL-SE).

OUTCOME

Reasonable agreement here that we should keep specific technologies outside the NZGOAL_SE and focus on principles that lead to selection of a suitable release space (however only 42% voted can this be considered "passed"?). There are some further spin off suggestions (summarised at top of thread). We may run follow up polls on each of these items (or split off into own thread for longer form discussion).

POSITIONS



42% of members have stated their position (10/24)

Strypey agreed:
I don't think general policy statements should advertise US-based corporations whose website uses a number of proprietary components (ie GITHub). If specific examples are given they should be chosen from among those that use only free code components

Edward Abraham agreed.

Gold agreed.

Dave Lane agreed:
I think we should agree that FOSS code should be managed using git, but the repository through which it is shared can be selected (or there can be multiple - see git mirror). Perhaps projects need to register their location somewhere centrally...

Cam Findlay agreed: For longevity of the policy, this is a good idea.

Sam Bonner agreed.

Brent Wood agreed.

Jason Ryan agreed.

Rob Elshire agreed.

T. Charles Yun agreed.

Collapse

ACTIVITY

JR **Jason Ryan**

> 61(a) using a version control repository like Github; and

Could offer other alternatives for balance (eg, Bitbucket) or, *preferrably*, suggest FOSS options like Gitlab which could be hosted in the .govt.nz domain, or they could use the NZOSS one until they stand up their own...

Liked by Gold, Sam Bonner and Dave Lane.

Cam Findlay

Perhaps the specifics on exactly what tech to use could be made available outside the policy (after all NZGOAL is all about licensing)? I note that the US Fed Govt are going through a public consultation on their FOSS policy and intend to have practical guidance and policy separate and referencing each other. Technology will change over time and the principles in the policy ideally can allow for this to have some longevity.

JR **Jason Ryan** in reply to **Cam Findlay**

Sure: I'm mostly against an ad for Github in a government policy document...

Liked by Don Christie, Dave Lane and Cam Findlay.

Dave Lane in reply to **Jason Ryan**

That sounds like a great idea 🙂

Dave Lane

For the record, we'd be happy to have people use the NZOSS gitlab instance on https://git.nzoss.org.nz

Cam Findlay

My thoughts here are that what we are really asking for is that code release occurs in a *publicly open repository* without imposing any particular technologies or locations.

Licensed code on a close repo doesn't promote discovery and reuse. That is a core principle of NZGOAL generally.

As mentioned, a separate set of practical guidance (like has been done with the NZGOAL guidance notes and webtoolkit) could start to suggest these things at a more practical level? (Perhaps this could actually be crowd-sourced from practitioners in this space or stewarded by a Govt Web Community of Practice). Just some thoughts.

As NZGOAL is primarily about licensing and has principles attached to it such as the Open Access principle, it's the right place to state that public access is preferable and not what or where that is.

In saying that, really awesome that NZOSS has an open repo space 👍

Roy Storey

Hi,

I agree with @camfindlay1 and @jasonryan about the wording and intention of policy and creating an advertisement for GitHub.

Implementors should be free to choose their open repository and its location and have as much control over it as they are willing to furnish and maintain.

BTW they are after us... https://jobs.github.com/positions/bea0d4a0-cb6e-11e5-9a96-a6303511a4e4

Cam Findlay

@roystorey interesting link, let's however keep discussions on topic 👍

Cam Findlay started a proposal: Code release should occur in a publicly open repository without imposing any particular technologies or locations. a month ago

Cam Findlay agreed: For longevity of the policy, this is a good idea.

a month ago

Dave Lane agreed:
I think we should agree that FOSS code should be managed using git, but the repository through which it is shared can be selected (or there can be multiple - see git mirror). Perhaps projects need to register their location somewhere centrally...

a month ago

Roy Storey

> Code release should occur in a publicly open repository without imposing any particular technologies or locations.

@camfindlay1 I wonder if accessibility is constrained. A publically open repository using an obscure technology is not as accessible as one using a commonly available one. Contributions are increased by simplicity - the only citation I have of this is https://www.youtube.com/watch?v=U8GBXvdmHT4&t=47m00s (a few seconds of listening to Matthew is all that is required...) I would be interested in having a better source to cite.

Cam Findlay

@davelane while I love git, suppose something else new and awesome comes out for version control? Other than being specific about licenses (MIT/GPL/whatever is decided upon) which is what NZGOAL is about, my proposal here is about being technology agnostic. However, could I suggest that we proposed code is at least "in version control" which is really the core principle of what we are perhaps asking for here?

"Perhaps projects need to register their location somewhere centrally..." <- all the yes for this and I think that is a bigger discussion perhaps beyond NZGOAL? I note the US Fed govt open source policy are looking to set up such a register for reusable open source things. See https://github.com/WhiteHouse/source-code-policy/blob/gh-pages/pages/Implementation.md

The US proposed policy is framed in a much more aggressive push to mandate new code release (that we haven't tackled, yet, in NZ). NZGOAL might be a solid stepping stone towards this (NZGOAL is focused on solid licensing principles and guidance).

Dave Lane

I'm strongly in favour of ratcheting up from "encourage" to "mandate", especially in terms of compliance with open standards, but also open source for all NZ gov't funded software development.

Roy Storey likes this.

Dave Lane

I'm also keen to see us stipulate a version control technology that is a) open (no barriers to adoption) and, b) widely used. At present, we're in the lucky situation that the most widely used is also open (as are next couple most widely used, e.g. mercurial and bazaar)... In future, I'd be keen to adopt the most widely used *and also open* technology, if/when something supersedes git.

Roy Storey likes this.

Roy Storey

What is the lead time on this *new* technology? Is the policy able to cope with editing this frequently? If not, slight ambiguity might be warranted to address longevity of the policy.

Can this be solved with Best Practice based on the policy? Best Practice documents may be more maleable than a policy that specifies the why / philosophy or chosing open source/version control/GPL etc...

Sam Bonner agreed a month ago

a month ago

Jason Ryan agreed a month ago

a month ago

Cam Findlay

@roystorey can we crowd-source these practice documents from practitioners on the ground or through communities of practice perhaps?

Roy Storey in reply to **Cam Findlay**

@camfindlay1 I'd like to say yes. Many government funded organisations are on GitHub and could contribute - if we are talking source control specifically, but this would extend to other informatics areas also.

Gold agreed a month ago

a month ago

Cam Findlay

@davelane would it be advantageous to stipulate in NZGOAL-SE that the version control technology (and software clients to access) conform to an open standard and/or be available as FOSS itself? Git, for example fits that pattern (along with others).

The open access principle might not work if a release is made, on a public version control system that requires some proprietary software client to access/track any contributions or modifications.

Pretty much, I don't mind if it's released on git, subversion, mercurial etc, as long as the tool is open and freely available for me to interact with the released FOSS software under NZGOAL-SE. 👍 I think that's what is important.

Dave Lane

I agree that I'm happy as long as the version control system is open source itself.

Brent Wood agreed a month ago

a month ago

Edward Abraham

I agree that it is great to separate mention of GitHub out from the policy. But there needs to be a how to guide, giving examples of how to satisfy the policy. Releasing software on Github is a common way for people to maintain an open source project. This choice (Github/Gitlab/Bitbucket), is about the broader considerations of maintaining the software, rather than the licensing per se.

Edward Abraham agreed a month ago

a month ago

Cam Findlay

@edwardabraham I started drafting something along these lines last year as practical guidance for Common Web Platform.

See https://github.com/camfindlay/opensource-nzgovt/tree/master/a_guide_to_better_code_sharing_on_cwp

Perhaps this could be built on into a more generalised set of FOSS guidance. Happy for it to be reused and crafted into some sort of help guide outside of NZGOAL-SE 👍

Edward Abraham likes this.

RE **Rob Elshire**

I agree with publicly open repositories. There should not be an imposition of non-FOSS tools either. Then there is the issue of discoverability. I think that is being discussed in a different thread.

Cam Findlay likes this.

Rob Elshire agreed a month ago

a month ago

Strypey agreed:
I don't think general policy statements should advertise US-based corporations whose website uses a number of proprietary components (ie GITHub). If specific examples are given they should be chosen from among those that use only free code components

a month ago

Strypey

As others have said, what the NZGOAL wording needs to do is tease out and make visible the principles that lead us to recommend GIT, and to endorse or not endorse specific platforms (eg GITHub/ GITLab). I'll attempt to sum up the principles that have fallen out of this discussion in the context box (top of the page).

Cam Findlay likes this.

Strypey updated the thread context a month ago

Cam Findlay in reply to **Strypey**

Thanks for the summary @strypey 👏

T. Charles Yun agreed a month ago

a month ago

The proposal has closed: Code release should occur in a publicly open repository without imposing any particular technologies or locations. a month ago

Cam Findlay published a proposal outcome:

Reasonable agreement here that we should keep specific technologies outside the NZGOAL_SE and focus on principles that lead to selection of a suitable release space (however only 42% voted can this be considered "passed"?). There are some further spin off suggestions (summarised at top of thread). We may run follow up polls on each of these items (or split off into own thread for longer form discussion).

a month ago

DC **Don Christie**

Hi, being independent of the likes of github and onshore is important for the initial releases. Others can mirror of they like.

Remember, there are *very* good reasons no-one uses sourceforge any more.

Strypey likes this.

Add advice for developers section

Started by **Igor Nadj** 2 months ago · Edited · Public

This is a long document that risks not being read by the people who should read it most, developers. I suggest adding a short section at the top of the document.

Suggested content:

Advice for Developers

Open sourcing your code is a great idea, it helps contribute back to the community, as well as improves your own codebase from contributions from the community. Additionally it lets you become part of the wider open source ecosystem which helps share modern ideas and tools to develop some really cool stuff. Many government agencies are already on board. You should aim to have your code open source by default.

Open sourcing your code involves creating a repository on a public platform like Github or Bitbucket. The most important thing to add to your repository is a README file with an overview of what your code does, and your License (see section 13).

ACTIVITY

IN **Igor Nadj** updated the thread context 2 months ago

IN **Igor Nadj** updated the thread context 2 months ago

Cam Findlay

So your advocating for a kind of TLDR; section?

See https://www.loomio.org/d/TfnmWEr5/alternative-version-control-repositories we are proposing not to include specific repo technologies and locations (since you mention some in your proposed wording).

Roy Storey

I would like to suggest that "Best Practice" documents would capture this advice and ways of working behaviours.

Jason Ryan and Cam Findlay like this.

Learning from overseas discussion

Started by **Dave Lane** a month ago · Edited · Public

I just came across this interesting article talking about the recent announcement in the US that much software developed by the US Gov't would be open source... Some very interesting points, some very similar to our discussion (the request to refer to "Free Software" as well as open source = our copyleft discussion), and some related to the use of open standards and the implications of signing the TPPA... http://thevarguy.com/open-source-application-software-companies/github-forum-highlights-public-views-open-source-us-gover
This may offer us some insights into implications here in NZ.

ACTIVITY

Dave Lane updated the thread context a month ago

Dave Lane updated the thread context a month ago

Cam Findlay

It is indeed an interesting time for this stuff.

The US Fed Govt public consultation on their open source policy ends 11 April and is being carried out using GitHub to accept feedback and direct policy wording changes for them to consider. I've read over it, lots of good stuff there for us to consider. See some of the feedback at https://github.com/WhiteHouse/source-code-policy/issues

What is interesting is, the US Fed draft policy covers mandate to share and process for procurement however notes they haven't covered the licensing aspect. For us, we are the reverse of this (NZGOAL should stick to it's licensing scope perhaps).

It might be interesting to reach out to those policy makers to see where there might be some reuse in the future? Could they "fork" and adapt what we're doing for their needs (USGOAL-SE anyone? 😀) and in future could we do the same?

Anyway, good to place what is happening in public sector open source here in a wider global context.

Dave Lane likes this.

Cam Findlay

Thanks for raising this @davelane 👍

Roy Storey

Thought I'd share this in case you haven't seen it. (Changing the max to 2500 highlights NZ creates the most commits per capita)

http://geeksta.net/visualizations/github-commit-map/

Strypey

Somebody shared the US gov 18F platform in the discussion on code discovery. Their "open source policy" has some good wording and structure that might a useful reference for the next draft of NZ GOAL-SE.

Source code/project catalogue (discovery tool)

Started by **Brent Wood** a month ago · Public

Source code repositories are good for devs - bit not for managers/policy people looking to make strategic decisions around software purchase or development. Like open data, delivery is easy - stick it on the web in a suitable format 🙂

The critical precursor - a discovery capability (not dev oriented) is required. This could be a standalone catalogue of relevant projects providing open source for govt initiatives, or perhaps incorporated in data.govt.nz.

However it is done, a suitable catalogue or discovery portal focussed on projects rather than code should be established & NZGOAL-SE should mandate its use.

This could be established very soon, & easily, & it could be populated with past & current projects.

Such a catalogue should link to the dev & home pages of software projects which have been reused (Silverstripe, Koho, Moodle, Alfresco, etc all spring to mind) as well as the local projects which reused them.

PREVIOUS PROPOSALS

Specific discovery tools should not be included in NZGOAL-SE as this would be out of scope (except to reference them in subsequent guidance notes).

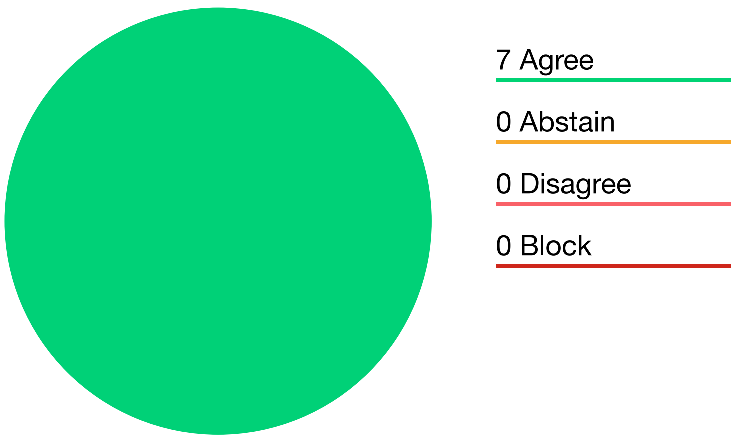
Started by **Cam Findlay** · Closed a month ago

There has been discussion here about code discovery tools and their place in NZGOAL-SE. I think we have some agreement that we should keep NZGOAL-SE scope to the licensing aspect and would like to take a quick poll on this to drive further discussion.

OUTCOME

Appears an acceptable and workable way forward on this appears to be to look at specifics in future guidance notes, with a more principled reference in the policy itself. 👍

POSITIONS



28% of members have stated their position (7/25)

Edward Abraham agreed:
Important there is some guidance of how to do this somewhere. But keep the core goal of getting software openly licensed firmly in focus.

Tom Clark agreed.

Dave Lane agreed:
I do think it's within the scope to say "the repository, which will be kept up-to-date must be registered with the provided central project directory"... without needing to define that solution.

Cam Findlay agreed.

Byron Cochrane agreed:
I concur with Dave Lane. But require entry in an audit-able and publicly accessible authoritative registry (without specifying software) as minimum requirement of publishing. Use similar language as in the versioned repository discussion.

Rob Elshire agreed.

Roy Storey agreed: What @davelane said

Collapse

ACTIVITY

Cam Findlay

Funny enough, this is mentioned in the US FOSS policy under consultation at the moment too. See https://github.com/WhiteHouse/source-code-policy/blob/gh-pages/pages/Implementation.md

US are going to be setting up https://project-open-source.cio.gov/ as a potential discovery and guidance space. Would something like opensource.govt.nz registry or something to that effect be desirable? I note in the NZGOAL-SE there is provision for setting up further guidance notes once the licensing framework is in play.

It would be good if the policy asked as part of the release process to list it on some central discovery space (who should maintain this?). However, the set up of this discovery space would have to happen outside this policy perhaps? I know this is something I've been thinking about for Common Web Platform and reusable FOSS code generated there.

Interesting discussion @brentwood 👍 keen to explore more and determine where this fits.

RE **Rob Elshire**

I'm keen to see this included somewhere. It is all well and good to have FOSS licensed software that exists somewhere in a version control system, but it has little value if it can't be found.

Roy Storey

I could see this working as a `nz.govt` file in the repository of the software in question, which is aggregated by `opensource.govt.nz` service. Examples for most languages and OS level distributions exist, think CRAN, CPAN, Rubygems, Debian APT, RedHat YUM, Windows Chocolatey, OS X Macports. Most of these have automated bots parsing the metadata files, all of which are subject to pull requests.

What is missing from these is "*for managers/policy people looking to make strategic decisions around software purchase or development*"

Policy wise: open source projects should include a <insert name> file in <insert name> format including adequate information to create an entry at <insert domain>.govt.nz

Cam Findlay

Would it be a workable position for NZGOAL-SE to say something along the lines of "to aid in discovery, released code projects should be submitted in the nominated discovery portal listed in any practical guidance notes"? The guidance notes then would be were we can work out the technical and practical details @roystorey mentions.

I note there is already a section for allowing further guidance notes in NZGOAL-SE draft (Section 7). The guidance notes are a more informal and updatable artefact than the NZGOAL-SE policy itself. Capturing the core principles in NZGOAL-SE and building out further practical guidance notes would help longevity and ongoing usefulness I'd imagine.

I think this might be similar to the thread on the principle of releasing in an open repository that we've had a vote on already. In that case we agreed on a principle and that we could address in specifics in practical guidance notes going forward.

Roy Storey likes this.

Edward Abraham

I agree that this should be outside of the policy itself. Directories of things need sustained effort to maintain (which means funding). Unless actively curated, they are often stale and so not that useful. I think the focus should be as narrow as possible (choosing and applying an open source licence), with the rest of the ecosystem around that being able to develop as required. Unless there is a funded mechanism, perhaps this shouldn't be included at this stage.

Liked by Byron Cochrane, Dave Lane and Cam Findlay.

Dave Lane in reply to **Edward Abraham**

We could look at repurposing something like the (open source) CSIRO Permanent URL service for data sources (see this instance of it http://registry.it.csiro.au/) ... whatever it is should be self-service registration, so that project coordinators can update the pointer to their projects in a central place themselves. It's all about getting the incentives right 🙂

Cam Findlay started a proposal: Specific discovery tools should not be included in NZGOAL-SE as this would be out of scope (except to reference them in subsequent guidance notes).

a month ago

BW **Brent Wood**

I think it is critical that DIA (& LINZ) realise that a delivery system without an effective discovery capability is pretty much a waste of effort. Given the lack of progress on geodata.govt.nz & limited support for data.govt.nz that has been apparent for the last few years, I'm far from convinced this is appreciated.

A policy or guideline recommending agencies follow a certain course, with no tool;s or framework to facilitate this, will, at best, result in a plethora of different, probably non-interoperable systems set up by each agency that tries to comply.

Providing guidelines, and a framework with tools enabling compliance is a big step in making it easy to comply with the guidelines, or at least removing significant barriers to adoption.

Much better than "here is what we want you to do - now go & work out how",
is "here is what we want you to do, and here is how you can do it"

NZGOAL-SE is not the place to contain the tools & docs, etc, just as NZGOAL points at CC rather than replicating everything there. It also uses CC to provide examples of how to follow the NZGOAL guiodelines - setting up links to online licences, etc.

BC **Byron Cochrane**

I would have to largely concur with Brent on this point. (And I am the guy nominally in charge of geodata.govt.nz!) While I agree with the general point that specific discovery tools need not be named, I do think that at a minimum, any software that falls under this guidance should be registered in a central registry. This register might be less than a full fledged catalogue. What metadata would be included in such a repository could be debated and evolve over time but it must include the uri to the versioned repository in which the software lives. (The maximum approach would be to recommend that any software that falls under this guidance should reside in a yet to be specified central versioned repository.)
The point for this would be that without at least a simple registry, "publishing" your software is not that meaningful. For the agency producing the code, this allow them to "check the box" and say yes, their code is published. For developers inside and outside of government, discovery is at least possible. An audit of activity would be possible against this authoritative list. Addition information could be linked to items in this register.
While it may not be the place of this document to recommend any particular solution to this problem, the need for such should be stated.

Cam Findlay likes this.

Edward Abraham agreed:
Important there is some guidance of how to do this somewhere. But keep the core goal of getting software openly licensed firmly in focus.

a month ago

Cam Findlay agreed a month ago

a month ago

Dave Lane agreed:
I do think it's within the scope to say "the repository, which will be kept up-to-date must be registered with the provided central project directory"... without needing to define that solution.

a month ago

Roy Storey agreed: What @davelane said

a month ago

Rob Elshire agreed a month ago

a month ago

Tom Clark agreed a month ago

a month ago

Cam Findlay

@brentwood did you have a position on this proposal poll (since you raised the original discussion point)? 😀

Byron Cochrane agreed:
I concur with Dave Lane. But require entry in an audit-able and publicly accessible authoritative registry (without specifying software) as minimum requirement of publishing. Use similar language as in the versioned repository discussion.

a month ago

The proposal has closed: Specific discovery tools should not be included in NZGOAL-SE as this would be out of scope (except to reference them in subsequent guidance notes). a month ago

Cam Findlay published a proposal outcome:

Appears an acceptable and workable way forward on this appears to be to look at specifics in future guidance notes, with a more principled reference in the policy itself. 👍

a month ago

Strypey

I agree with flagging the need for code to be discoverable in NZ GOAL-SE, and detailing implementation specifics in supplementary notes.

To address implementation, are we talking about a directory for free code software in general, or the subset of free code software used in government? Even if it's the latter, creating a new system specifically for "open government" software seems like an unnecessary duplication of effort, considering there are already a number of existing general directories:

- Free Software Directory: in existence since at least 2002 and maintained by the FSF. Structured as a semantic wiki that anyone can submit new entries or updates/ corrections to, with an editorial group checking submissions for accuracy before approving. Database rights are released under GNU FDL 1.3 (or later). Note: will only list software that can run on a free code OS.
- Open Hub: launched in 2006 as Ohloh, Sold to Geeknet is 2009, who sold it to Black Duck in 2010. Like the FSD, OpenHub is a wiki, but it also features automated tools that search across code repositories. Database rights are released under CC-BY 3.0.
- FreshCode: An attempt to re-implement FreshMeat/ FreeCode(archived). It accepts logins using OpenId, and can accept automated updates from projects, using "JSON-based database exchange feeds" (I presume). Database rights are released under CC-BY-SA (unclear what version).

Maybe the solution is a cross-government project to make sure all relevant software is included (correctly) in one (or all) of these, identified with a general tag (something like "used in government"), and categorized using more specific tags (eg "maps" or "CRM")?

Cam Findlay likes this.

Cam Findlay

Thanks @strypey - an interesting idea, perhaps one reason I could think of for a government open source discovery portal might be around discovery of both *proposed* and "finished" applications and code, emphasis on proposed (let's also be honest, open source is never truly "finished").

18F in the U.S. have this site for their digital services, gives a good indication as to the stage things are at for a project. I can find out details of who to contact for a project in the discovery stage and perhaps look to collaborate. Something like this might be useful to be checked as a first port of call when thinking about building something new for government (for the technically minded, it uses a Yaml metadata file in the referenced repos to keep dash data up to date, that's been mentioned previously and not something to solve now).

Maybe Agency B finds out that Agency A is in the discovery phase of a new open source GIS mapping application which is pretty close to what they need (through the portal, Agency A has indicated to others their intentions to built such a thing). They should as a matter of course look to work together as opposed to building it in isolation and THEN registering it on a discovery space only to find they built the same things.

I guess the principle to extracted here, regardless of the discovery space being an existing community one or a specific nz public sector one is the "Agencies should do an environment scan of existing open source software discovery portals to ensure they are not building something already built".

Strypey likes this.

Strypey in reply to **Cam Findlay**

Absolutely. There's a parallel here to academic research projects, where the first thing you do before you start your research is a literature review, to make sure as much as possible that you're building on existing research, not duplicating it. In fact, this analogy might be a useful way of explaining the relevant practical advice in NZ GOAL-SE. A classic real world example for software would be CMS. It would be a crazy use of public funds for a government department to build their own CMS from scratch (or lease a hosted proprietary one), rather than building an special features they need as modules or plug-ins for an existing free code CMS like WordPress or Drupal (or Totara in @donchristie 's Customs example). Indymedia learned this the hard way (as mentioned elsewhere).

Cam Findlay

See https://github.com/opendatanz/nzgoal-se/pull/11/files

When would you pick GPL?

Started by **Cam Findlay** a month ago · Public

In this thread, we'd like to explore the situations in which GPL is a good, practical choice of license for the release of *new* open source code in government.

For context please read over the thread discussing the current draft defaults and options.

Please keep the discussion on the merits of GPL and when it is practical to select it.

The outcome of this thread is we'd like to generate some consensus on useful guidance that helps users make good license choices in the right situations.

We'll also have a thread along the same lines for the MIT discussion, please feel free to comment in both. 👍

ACTIVITY

Dave Lane

You use the GPL when you want people to contribute to the project in the knowledge that their contribution will never be used to profit without their being able to benefit on the same basis. The GPL focuses on protecting the rights of the user, and assumes that any user might become a developer. This is most appropriate for end-user applications. Consider AGPL for end user centralised web applications/SaaS models, as the GPL v3 does not protect user rights in this situation (as the "distribution" provision of the GPL v3 is not triggered in that case).
If you are trying to get businesses and/or other software projects to adopt a library (to achieve a standardised API, for instance) you might license under a more permissive license like MIT to allow incorporation of the library into proprietary applications.

Liked by Sam Bonner, Rob Elshire and Strypey.

Cam Findlay

Thanks for that feedback @davelane - could you please post your MIT comment over at https://www.loomio.org/d/0J9qHdsT/when-would-you-pick-mit- I'm going to use these threads as a capture of collective wisdom on the aspects of these two license approaches 😀

So just to see if I get your point correctly, for GPL is that it is appropriate to pick this license when the application is a self contained application (may there is a better word for this?) that an end user (be it govt, business, citizen) can use as is or customise (with the understanding that distribution of this customisation needs to be GPL too).

For example, a Learning Management System such as Moodle (which is GPL) can be installed and used even altered at a code level. It is unlikely this application would become the base source code of a derivative commercial application that is further on sold. It's used for the purpose it was developed for, in this example, Moodle allows anyone to setup an e-learning platform for online learning.

RE **Rob Elshire**

To riff on Dave's example above: The GPL can provide a mechanism for playing nice between NZ government funded entities. Given our competitive funding mechanisms and that some institutions are using similar (same) technologies, a share alike license makes space to work together while keeping any actor from taking advantage of the others.

In my view, it is not just a 'stand alone' application for which the GPL is appropriate. It could be used to foster an ecosystem of interrelated applications. This could smooth the political issues around otherwise competitive organisations and allow them contribute to the development of applications that they would benefit from.

Dave Lane likes this.

Dave Lane in reply to **Cam Findlay**

Regarding your question - if the goal is to optimise a contributing developer community for an end-user application, GPL seems a very good license choice. I think GPL is good for the whole stack, personally, and prefer to prioritise the rights of the user over a hypothetical "developer" who (e.g. in business) wants to exploit someone else's code for their own gain at the expense of the user (which is one of the circumstances facilitated by a non-share-alike FOSS license like MIT) creating an implicit developer-user dichotomy. The GPL fundamentally assumes that every user can and should have the ability, without requiring permission, to be a developer, too, removing the developer-user distinction. This freedom to afford oneself the role of developer rather than just user is equally relevant for infrastructural resources (e.g. libraries) where the typical user is actually a developer farther up the technology stack.

GasparI in reply to **Dave Lane**

LGPL is perfect for libraries - can use it anywhere and also the community benefits. Win-win. http://www.gnu.org/licenses/lgpl-3.0.en.html

Strypey likes this.

Edward Abraham

A bit of an anecdote: we developed a website for MBIE that displays regional information in map and time-series form, see http://webrear.mbie.govt.nz/summary/new-zealand. We are wanting to open source the framework so that other data sets can be displayed in the same way. MBIE have indicated to us that a GPL licence would be their preference. I think this is primarily so that other developments get contributed back to the project. (cross-posted).

Strypey likes this.

DC **Don Christie**

That's great. MBIE are not the only agency that has plumped for GPL for new work. I am fairly sure Archives NZ did as well.

Strypey in reply to **Edward Abraham**

If this MBIE mapping software is a server-side package, then as @davelane has said, AGPL(v3) would be a better choice than vanilla GPL, because otherwise a modified version of their software can be run on a server without triggering the copyleft provision, and therefore without obliging the modifiers to share their code.

Strypey

One advantage of using GPL or AGPL is that it mostly gets rid of the need to consider copyright ownership. When a person modifies and distributes your code, the copyright for the new code stays with them, but unlike with a non-copyleft license, doing that triggers the copyleft provision. This means their modified version is automatically licensed under the same license as your original, which means you can safely use, modify, or redistribute it under the terms of (A)GPL, without needing their permission. The only times copyright ownership matters in practice with copyleft licenses AFAIK are:

- you want to reissue the code under a different license. If you don't own the copyright to the whole codebase via a copyright assignment agreement, you'd need each external contributor's consent to include their bits of code in the re-licensed version
- you want to enforce the license. Copyright law only allows the copyright owner to press copyright violation charges. Small GPL projects lacking the resources to enforce their license sometimes give copyright ownership to the FSF to enforce on their behalf.

BTW IANAL and I assume those involved in this drafting process have made contact with Eben Moglen or someone from the Software Freedom Law Centre, who can confirm or dis-confirm anything those of us who are not lawyers are saying?

Russell MIchell

The very fact there is so much discussion around "which license" suggests it is not a mere trifle in deciding which to use. OSS developers already know this, but others new to the area, e.g. I.T. professionals having worked with pay-to-license software for the entirety of their careers, perhaps would not.

I think the development of an online tool that helps users in their decision is one way forward here. See http://choosealicense.com/. Luckily the site's codebase is itself OSS and available here: https://github.com/github/choosealicense.com and could therefore conceivably be modified for NZGOAL-SE's uses.

GasparI and Strypey like this.

GasparI in reply to **Russell MIchell**

It's a great site, thanks for the link!

Strypey in reply to **Russell MIchell**

True. If picking a license was simple, all NZ GOAL-SE would need to say is "choose a free software license" and collate a list from the lists provided by the FSF or OSI 😉

At the risk of going slightly off-topic, the CC license chooser is a really good model. Some folks from FSF, OSI and/or other free code organisaions like Software Freedom Conservancy, Software in the Public Interest (Debian parent body), Linux Foundation, or Apache Foundation, might be keen to collaborate on creating a similar license chooser for software.

When would you pick MIT?

Started by **Cam Findlay** a month ago · Public

In this thread, we'd like to explore the situations in which MIT is a good, practical choice of license for the release of *new* open source code in government.

For context please read over the thread discussing the current draft defaults and options.

Please keep the discussion on the merits of MIT and when it is practical to select it.

The outcome of this thread is we'd like to generate some consensus on useful guidance that helps users make good license choices in the right situations.

We'll also have a thread along the same lines for the GPL discussion, please feel free to comment in both. 👍

ACTIVITY

Dave Lane

An appropriate case for selecting the MIT license is when you want to propagate a new API or software library that could be incorporated in to many applications, for example a new encryption mechanism. The MIT license minimises the barriers to adoption by proprietary project developers (because it places no limitations on their ability to exploit their work) and most FOSS projects (I'm not sure if GPL'd projects can adopt MIT licensed components, although I believe in that case the GPL and MIT are compatible. Keen for verification of that point).

Cam Findlay

Thanks for reposting @davelane - agree that it would be good to get verification on your point of GPL projects being able to make use of MIT parts.

So to clarify your MIT choice case, you are saying that libraries and component code (again perhaps there is a better agreed term for this type of code) that is used to extend existing projects or; on which new projects (commercial or not) might be built upon, in order to promote reuse of standards expressed in those libraries, then MIT is a great choice.

GasparI

LGPL is perfect for libraries - can use it anywhere and also the community benefits.
I would be really interested to hear a use case where an MIT license would be 'better' than LGPL for the community.
http://www.gnu.org/licenses/lgpl-3.0.en.html

Russell MIchell

I have x-posted the following in the GPL thread also:

The very fact there is so much discussion around "which license" suggests it is not a mere trifle in deciding which to use. OSS developers already know this, but others new to the area, e.g. I.T. professionals having worked with pay-to-license software for the entirety of their careers, perhaps would not.

I think the development of an online tool that helps users in their decision is one way forward here. See http://choosealicense.com/. Luckily the site's codebase is itself OSS and available here: https://github.com/github/choosealicense.com and could therefore conceivably be modified for NZGOAL-SE's uses.

Cam Findlay and Strypey like this.

Cam Findlay in reply to **Russell MIchell**

Thanks @theruss I'm a fan of that site, keeps the decision tree of selecting suitable a license simple and effective. The descriptions and bullet point overviews are very useful (not to mention it's both content licensed under CC-BY and code is open under MIT) 👍

Strypey in reply to **Dave Lane**

> I'm not sure if GPL'd projects can adopt MIT licensed components

While there may be some variation in compatibility between GPLv2 and GPLv3, the GPLv3 wiki has a list of licenses that are compatible with GPLv2, and it includes both licenses commonly referred to as "MIT" (Expat and X11).

Also, Karl Fogel writes in Chapter 9 of his book "Producing Open Source Software":

> "While the Apache License 2.0 has the advantage of containing some explicit defenses against misuse of software patents, which might be important to your organization depending on the kind of project you're launching, the MIT license is fully compatible with all versions of the GNU General Public License, meaning that you can distributed, under any version of the GPL, mixed-provenance works that contain MIT-licensed code. The GPL-compatibility situation for the Apache License, on the other hand, is more complicated — by some interpretations, it is compatible with GPL version 3 only. Therefore, to avoid giving your downstream redistributors the headache of having to read sentences like the preceding ones, I just recommend the MIT license as the default non-copyleft license for anyone who doesn't have a reason to choose otherwise."

Remember that by definition, "permissive" non-copyleft licenses contain no rules about what license applies to redistributed or modified versions. This is one of the qualities their proponents refer to as "permissive". That being the case, my understanding is there are a number of ways this could work for Project "Foo" (GPL license), and Component "Bar" ("MIT" license). Keep in mind as always that while I make a determined effort to get my facts right, IANAL, and anything I claim about software licensing should be checked against more qualified sources (eg Software Freedom Law Centre).

Firstly, what could happen if Project "Foo" (a graphical desktop application) incorporates a verbatim copy of Component "Bar":

- Project "Foo" identifies Component "Bar" as an external "dependency" in its packaging instructions. This means that on GNU/Linux (and most Unix systems), when a user installs a package of Project "Foo", the latest version of Component "Bar" in the same software repository will be downloaded and installed too (unless its already installed). If Project "Foo" package their own binaries of their application (eg to support use on Windows or MacOSX), they can include the most compatible version of all non-copyleft dependencies like Component "Bar", as long as a copy of the license for each dependency package is also included. *Note:* they can bundle copyleft dependencies too, but on the top of the copy of the license (or a link to one), they must also include a copy of the source code (or a link to one). Non-copyleft licenses don't require this.

- Project "Foo" copies the source code of Component "Bar" into their own repository, in its own section, and distributes it under the terms of the "MIT" license supplied by the developers of Component "Bar". A copy of that "MIT" license must be included with the code. *Note:* Project "Foo" will usually identify Component "Bar" on their website and in their help files, and give credit to Component "Bar"'s developers ("attribution" as required by CC-BY) , but they are not legally required to do so by any of the licenses commonly known as "MIT". Thus, an "MIT license" is not fully equivalent to CC-BY.
- Project "Foo" copies the source code of Component "Bar" into their own repository, and distributes it under the terms of the GPL. *Note:* this doesn't stop anyone using, modifying or redistributing it under the original "MIT" license, and as above they must bundle the original developer's license with the code anyway, so while this is legal, in practice there's little point to it.

Now, what could happen if Project "Foo" incorporates a modified copy of Component "Bar"? In each case, for as long as any "substantial portions" of Component "Bar" remain, a copy of the original "MIT" license must be distributed with it.

- Project "Foo" copies the source code of Component "Bar" into their own repository and modifies it. They distributes their modified version under the terms of the same "MIT" license and submit their changes back as "patches" to the original developer, who may or may not incorporate them into the next release of Component "Bar".
- Project "Foo" creates a "fork" of Component "Bar". They copy the source code into a new repository, give it a new name, modify it, and maintain this new version as a side project. They might do this to make maintenance easier if Component "Bar" was "orphaned" (not being actively maintained), or the original developer of Component "Bar" was not interested in incorporating their changes. The forked version could be licensed under the original "MIT" license, or under GPL (or any GPL-compatible license for that matter).
- If Component "Bar" was part of a larger program, Project "Foo" could rewrite it as a library that could be linked by their program, and also by other programs. In this case, LGPL is a third license option that might be chosen (for reasons discussed in another thread) as well as either of the two above.
- Project "Foo" copies the source code of Component "Bar" into their own repository and modifies it. They distributes their modified version under the terms of the GPL. This makes it harder for the original developer to incorporate the modifications into a future release of Component "Bar". They would have to get permission from Project "Foo" to use their modified code under the "MIT" license, or they would have to relicense Component "Bar" to GPL. *Note:* under the terms of a "permissive" license, it's perfectly legal for Project "Foo" to do this. But if they are going to maintain a modified version of Component "Bar" themselves they are better to fork the project, as described above, to avoid confusion.

EDIT: Sorry about the super-long posts. Please let me know if you'd rather I post the long form pieces to my own blog, and publish a TL;DR here with a link to it.

Code forking (section 29-30)

Started by **Cam Findlay** a month ago · Public

Reading over the section 29 and 30, I think what is being discussed here is avoiding *diverging forks* and reads like "forking is bad". As someone that works in open source code daily, my understanding of the term "fork" is a positive thing. I can take a fork (a copy of the code) and make my changes without having ask and then from my fork offer code back to the original project (after discussion of my feature change with the original maintainer of course!).

I think that 1) perhaps a better phrase than "Code forking" for this section might make sense (any idea?) and; 2) this sections purpose seems to be about code contributions back to existing open source projects and; 3) this section is trying to guide agencies to use the canonical versions of code repositories rather than diverging forks of released open source projects.

I'd love to hear some comments around this. 👍

PREVIOUS PROPOSALS

Principles about the licensing aspects of contributions and Contributor License Agreements should be included in NZGOAL-SE

Started by **Cam Findlay** · Closed 19 days ago

NZGOAL-SE is all about dealing with **licensing**. One aspect of licensing that I think needs to be addressed in the policy itself is that of dealing with how licensing works when a member of the public or another government agency staff member makes a contribution to a project once it has been released on open source terms.

I don't think it's currently clear in the wording of the draft. There is mention if *modifying* the code (presumably in a copy of it) to use the existing licensing from the original project, however raising a pull request (a proposal of a fix/improvement to code) might be on the original code repository not a copy.

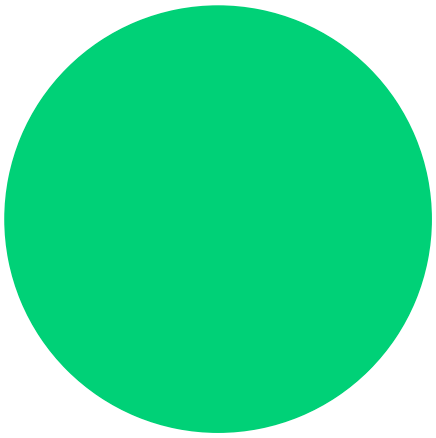
This proposal suggests that some principles about how you might deal with considering contributions to projects once released may have a place in NZGOAL-SE in particular:

1) Projects should include a README or CONTRIBUTING file that explains how contributions will be incorporated into the code base and under what terms.

2) A clear statement should be included in the above file about **copyright assignment.** Does the contributor retain copyright and license their contribution to the project OR does the contributor assign copyright to the project maintainer and in return, gain the bundle of rights to use/modify/redistribute under the license already in the project? When would you chose either approach or should one be the recommended?

3) Guidance on when a Contributor Licensing Agreement (CLA) should be signed by a contributor (guidance notes could explain how to do this and the when and why should be in the policy).

POSITIONS



1 Agree

0 Abstain

0 Disagree

0 Block

3% of members have stated their position (1/37)

Cam Findlay agreed:
I'd be keen to see something along these lines (even just a set of principles and then the rest in guidance notes).

Collapse

ACTIVITY

Cam Findlay

This section might also cover what the license/sub-license situation is when someone either submits or accepts code contributions into release government open source projects.

If someone makes a fork of a code repository with an existing license then that license applies to that fork or copy of the code. If I then write some new code on top of this, I'd be the first owner of the copyright of that new code.

Then I offer that new code back to the original project from where I took my fork from, am I actually offering that back under the same license of the original code repository OR does my bundle of rights under NZ Copyright Act still apply in tact (not licensed)?

In the spirit of open source, I'd hope that the intention is that the contributor is offering open source licensed code back, however legally I'm not sure how this works. I'd also hope that some sort of complex rights assignment process wasn't involved as I can see this working against accepting fixes from the public into govt code (a benefit of open source code).

Would be great if anyone knows the finer points here to chime in. ❓

DC **Don Christie**

Hi

1st rule, which I think was covered in the draft, if you are releasing code back to the project, new or otherwise, use the same open source licence as the project, regardless of your preference. Catalyst prefers the GPL but we are delighted to be able to contribute BSD licensed code upstream to the likes of OpenStack and Silverstripe. And why not, both platforms have given us great value.

The GPL is clear on modifications. You own the copyright but if you you have simply modified an existing program you *and* you chose to give it to someone else you need to give it to them under the GPL licence.

If you have developed something new (like a Drupal module or plugin), you get to decide what licences to use. You can release code under multiple licences proprietary and open source. That's what Alfresco and other projects to.

In this example, the Customs plugin could have been released under any licence. But Totara is GPL (based on Moodle) so it would have been a bit shit to release under a different licence.

http://catalyst.net.nz/news/nz-customs-service-gives-back-totara-plugin

What many projects ask contributors to do is sign a contributor agreement. These cover the fact that you really do have the right to contribute code and that you waive any patents you have that may cover your code. Some CAs also ask you to assign copyright to the governing body. The latter type do this for 2 reasons. Firstly it is easier to defend the entire work against licence infringers and secondly so that they can do the multi-licence thing. Have an open source "community" edition and a proprietary version as well.

Cheers
Don

Strypey likes this.

Cam Findlay

What I'm drawing from above comment is that for a government released FOSS project, contributions back from the public might be easier to manage if the project was GPL. As if I modify the code and offer a patch back it would be under GPL too so the copyright bundle of rights licensing aspect would then be clear. The author retains the copyright and the code would be offered back under GPL for further redistribution.

I'm unclear (personally) how this works in the MIT situation (hence my dialogue here to get some perspectives and input from wider community experience). If for example, I accept the original MIT license of a codebase, then I modify it with new code, my new code would need to be itself licensed in a way to offer it back as a patch to the original MIT licensed project. This could be complex and a blocker for others contributing into released government FOSS projects. In this case would guidance be required here?

Additionally, @donchristie I think we are on the same page about respecting the license of already released FOSS projects and communities 😀 .

Further, guidance on a Contributor Licensing Agreement (CLA) would be useful, in particular assuring the contributor actually has the copyright in order to contribute the code. This could reside either in a set of additional Guidance notes (as has been raise in other threads) or as a suggested inclusion in the NZGOAL-SE (at this point unsure which way would be best so up for discussion).

Has anyone got other comments around the topics of CLAs or forking of code as a positive or an undesirable thing or the legal aspects of offering code contributions?

Cam Findlay started a proposal: Principles about the licensing aspects of contributions and Contributor License Agreements should be included in NZGOAL-SE 20 days ago

Cam Findlay agreed:
I'd be keen to see something along these lines (even just a set of principles and then the rest in guidance notes).

20 days ago

Strypey

Through all of these, discussions, valuable as they are, I keep thinking we're all talking somewhat in the abstract. This is why I've brought up Koha a couple of times, as it's one I've read a bit about, but I am here as a software freedom *activist*, I can't comment from an active developer perspective. So, it's great to see @donchristie mention the Totara example, and more such case studies from participants who are developers would really help to ground the discussion.

One case study I can comment on as a community developer (not a software developer) is Indymedia, the network of open-publishing, activist news websites founded in 1999. Benjamin Mako Hill (an Indymedia techs also involved with Debian and the FSF) wrote a good article called 'Software (,) Politics and Indymedia' about the open source development dynamics that drove Indymedia's multiple CMS projects. It summarizes the way that Active, the original CMS software used in Indymedia, was forked and adapted into eight different projects, which shared a very similar user interface, with totally different things going on under the hood.

He quotes the Jargon File on the meaning of a 'fork':

> “what occurs when two (or more) versions of a software package’s source code are being developed in parallel which once shared a common code base, and these multiple versions of the source code have irreconcilable differences between them.

Of course, this was all written pre-GIT, and @camfindlay1 's positive understanding of forking reflects changes in practice that have been made possible by the radially different way GIT works, compared to previous versioning systems. Before GIT, merging versions that had wandered off in different directions was at best painfully tedious, and at worst, outright impossible. Beyond a certain point forking became irreversible, and overly fragmented projects were at risking of dying, even though there may still be many developers working on different forks and plenty of users keen for new versions. Because of this, open source projects tended to discourage forking unless there was a very good reason, and instead allowed developers space to scratch their own itch by having complex versioning conventions with things like stable and testing branches (as Debian still does).

TL;DR what 'forking' means by default will tend to depend on how long the developer interpreting it has been in the game. I think confusion can be avoided though, as long as NZ GOAL-SE has one phrase for GIT-style forking, and one for divergent forking in the older Jargon File sense, defines them in the document, and uses them consistently.

The proposal has closed: Principles about the licensing aspects of contributions and Contributor License Agreements should be included in NZGOAL-SE 19 days ago

Strypey

Just a note on CLAs since you brought them up in the proposal (which I agree with BTW but it closed before I managed to indicate that 😉 If you're using a copyleft license, it's not as crucial, because just by modifying and distributing your code they're licensing it under the original license. If you're using a non-copyleft license a CLA is now considered essential to make sure the contributor is actually licensing their code under the original license. This protects the project from litigation trolls somehow acquiring copyright ownership over some part of the code and using it to extract money from the project, or more commonly, end users of the software (eg the SCO case against Linux). It also helps protect the project legally if someone has deceptively submitted code they don't hold the copyright on (and its not under a compatible free code license).

As for copyright assignment, one argument for this is that you can only assert a copyright claim over code when you own the copyright, but if the government holds the copyright over the core code of a package, just enforcing that would usually be enough to school anyone violating the license. That presumes the developers are government employees who sign away copyrights on what they do at work as part of their employment contracts (don't know how common this is in NZ public service), or contractors who sign the copyright over the to the government as part of the commissioning contract (which arguably would have been better in the case of Koha). An argument against it is that it risks future managers of the stewardship body changing to a proprietary license (see the thread on "open core" business strategies), but this can be dealt with by a clause in the CLA that says any version of the contributors code must always be licensed under a free code license, or even specifying what kind of license (eg copyleft or non-copyleft) or even which license (with an 'or later' condition that covers what should happen if that licenses was ever retired).

There is a mix of practices in open source communities around copyright ownership, with some projects that don't require copyright assignment:

- Linux kernel
- Ubuntu: http://www.ubuntu.com/legal/contributors
- Eclipse: http://www.eclipse.org/legal/CLA.php
- Nylas: https://www.nylas.com/cla.html

... and some projects that do require copyright assignment

- VoltDB https://voltdb.com/contributor-license-agreement/
- Apero: https://www.apereo.org/licensing/agreements

Cam Findlay

See https://github.com/opendatanz/nzgoal-se/pull/10

DC **Don Christie**

Hi

I am not a developer or a user of github...can I see a version of the document as it stands right now?

Thanks
Don

Cam Findlay

Hi @donchristie you can read the human readable version of the current state at:

https://opendatanz.github.io/nzgoal-se/

Note: the review and release section is still to come (though the revised tree is up).

DC **Don Christie**

Hi

Thanks Cam. I presume this link:

https://opendatanz.github.io/nzgoal-se/review-and-release-process/

is the one to follow?

Cheers
Don

Cam Findlay

@donchristie see https://www.loomio.org/d/iqPA14xe (splitting this discussion off to separate thread)

Section 1 & 2 (Purpose) - discussion

Started by **Cam Findlay** a month ago · Edited · Public

I'm going to start picking through my notes that I made after reading over the NZGOAL-SE.

In this section:

1) Some of the bullet points I think could be merged and made more concise.
2) The intro mentions that Government agencies "often own the copyright in the software ... that is developed for them" - I'd like to get an idea from industry how often in practice this is contracted out of.
3) I think it's good to tie the policy to the NZ Govt ICT Strategy (Section 1f) out of the GCIO office (I write about this in a research paper I put together last year), perhaps though we don't tie the exact dates as mentioned in the draft as this would quickly date the NZGOAL-SE policy as the ICT strategy gets revised (and it's good that strategies do get revised!).

Other than those points for me I'm generally happy with it.

Any other comments (generally or on my feedback)?

ACTIVITY

Cam Findlay updated the thread title: Section 1 & 2 (Purpose) - discussion a month ago

Cam Findlay updated the thread context a month ago

Dave Lane

Cam - Egressive (my former company) had copyright ownership as a standard term in our terms of engagement. It was done to ensure we had the ability to open source the software. We granted a non-exclusive license to our customer for the versions of software we provided them, with the ability to release that specific version of the code under the license of their choice (within, of course, the context of other licenses on which the software might depend).

Dave Lane in reply to **Dave Lane**

For the record, I make those Ts and Cs available to anyone to use/repurpose (under CC-By) here: https://davelane.nz/terms

Edward Abraham likes this.

BC **Byron Cochrane**

It seems to me that there are really two questions here that this document should provide guidance on. First is the question of copyright. Should the government retain copyright on software developed on their behalf using public funds? Second is the issue of licence. The guidance here of course should be that, regardless of who retains the copyright, the software should be openly licensed. The second question is obviously the focus of this work, but I would think that some guidance on the first would be necessary. I am not sure where I land on question one. Perhaps it is a case by case situation. Dave's arrangement in Egressive seems satisfactory to the spirit of what we are trying to accomplish. It is similar to the arrangement we made with ZNO for 3D building data while I was with the Canterbury SDI - we paid for ZNO to cc licence the data and not for the data itself. In that case it removed the need for us to manage the data while making it freely available to the public. The same arrangement could be useful in software products. However, this arrangement does sacrifice future control to the contractor and some companies may not manage this so well or take advantage of it.

Cam Findlay likes this.

Cam Findlay in reply to **Dave Lane**

@davelane , a thought experiment. 🤔

If there had been a clear policy for public sector FOSS licensing and release of the type under proposal in NZGOAL-SE when you drafted your standard contract, might this have changed the copyright assignment clause as a supplier (that is, the contracting out of the commissioning first ownership of copyright)?

What if the commissioner of the work asked to retain first ownership copyright?

Cam Findlay

@byroncochrane I think the following sections in the NZGOAL-SE draft my cover your *question one* proposed above.

Sections 9-12 & 31-33.

Have a look over these and perhaps let's start a thread specifically about those points. That will help in the final collation of the feedback 😀

Dave Lane in reply to **Cam Findlay**

@camfindlay1, we only had a couple customers even question our terms in 14 years. It wasn't an issue for most. Given that we were working non-gov't entities as well, I suspect we probably would've used the same terms even with NZGOAL-SE...

Cam Findlay in reply to **Dave Lane**

In that case would you still discuss with a govt entity the option of making the project code reusable under a FOSS license (regardless of who actually is doing the open sourcing)? 😀 And might you point towards something like NZGOAL-SE to help clarify to that client?

Dave Lane

@camfindlay1 of course we'd use NZGOAL-SE to lend credibility to our request to ensure any code we developed could/would be open sourced (usually GPL'd) - we didn't do much work that didn't fit that requirement.

Cam Findlay likes this.

Edward Abraham

(Note: comment below now in its own thread)

This policy should be closely linked to the IP section of the ICT procurement policy. That document has a decision tree which outlines when a supplier should retain copyright. Ideally, the procurement policy is modified so that the decision tree leads to a recommendation of an open source licence. I guess this is about recognising the bigger context that open source licensing will sit inside.

We successfully use the ICT procurement policy last year to help us retain IP in software we developed for a government client. The decision tree and templated contract clauses were extremely useful.

PS the ICT procurement policy is no longer online (except as a PDF). Is it still active?

Cam Findlay

@edwardabraham could I ask that you start a new thread along the lines of what I've done here (to discuss one particular section of the draft) and cross post your above post.

I think it could be useful to open up further focused discussion around the points you raise and be helpful for the final collation of the feedback during the consultation period. The thread could look to discuss sections 9-12 which deals with the IPR/procurement aspects the proposed NZGOAL-SE touches on. Thanks 😀

Edward Abraham in reply to **Dave Lane**

Thanks for this Dave. Great to hear that your bold, principled approach has been generally accepted by your clients. May well refer back to this ...

Business Case for each recommendation

Started by **Cameron Shorter** a month ago · Public

Firstly, congratulations on developing this policy. It will help agencies be much more effective in their use of Open Source.

Re application: I suggest this document will be more useful if language is used which helps readers define the business value of one recommendation over another.
For instance, giving back a code fork costs time and effort which equates to money. To be done properly, having a feature incorporated back into an open source baseline will typically add 20% to 40% extra effort coordinating with the community. Will this effort cost more than the long term benefit?

Recommendations should be included in this policy to help readers construct answers to these types of questions.

At risk of my comment being dismissed due to blatant self promotion, in my job at LISAsoft we have been supporting the implementation of Open Source for governments for decades, including providing health checks to Open Source development strategies and writing Open Source Policies. I'd love to be approached to apply the same to this project.

Warm Regards, Cameron Shorter < c a mero n DOTshort er A T gmai l .c o m>

ACTIVITY

Cam Findlay

Thanks for the comments and welcome to the conversation @cameronshorter 😀

Interesting perspective and some clarifying questions.

When you say "...define the business value of one recommendation over another." are you referring to selection of licenses i.e. MIT or GPL decision?

Do you have some data to support the 20-40% extra time mentioned? (No problem if this is more based on experience though 👍).

Is this assuming working on the code in private and then releasing as a patch to the project or working iteratively in an openly collaborative way during the code development process itself? In my experience as a developer working with open source, I've found the later approach not to add significant development time to project work and I could imagine the former might add time as mentioned.

Interested in exploring this from many perspectives 🤔

Cameron Shorter

Hi camfindlay,

Re "defining the business case", I'm referring to the multitude of business justifications that projects face when selecting one business strategy over another. Should I use one technology or another? Should I use open source or proprietary? Should I work collaboratively as part of an Open Source community and contribute to the development of community features which are not relevant to me, or should I just focus on the one feature I want improving, then throw the code back at the community and hope they take it? Should I use MIT or GPL license? Etc, etc.

The 20%-40% is approximate, based on my personal experience working with many Open Source projects. Sorry, no empirical metrics to back that up.

Cam Findlay likes this.

Cam Findlay

Thanks for the response @cameronshorter appreciate the clarification.

Cam Findlay

Clear business cases and guidance on practical matters are a good to have. What you suggest here would make great guidance notes around NZGOAL-SE.

There has been some other discussions along these lines recently in other threads that the community have agreed would warrant later some practical guidance notes being created.

What do you think? 😀

The key thing here is to keep in mind is the scope of the NZGOAL-SE framework is focused around the FOSS licensing processes specifically.

Dave Lane

My concern here is that the "business cases" are presumably focused on the case from the bespoke software service provider perspective rather than the gov't. I think the key thing here is to remember that the customer, ultimately, is the taxpayer, and we must prioritise their interests above the commercial interests of individual suppliers. Otherwise, this system is designed to exploit the commons for private benefit rather than enrich the commons for public benefit.

Strypey likes this.

Strypey in reply to **Dave Lane**

I agree that public benefit needs to be central to NZGOAL-SE itself, being an advisory to government departments and other public agencies. I think there is value in what @cameronshorter suggests, which I interpret as a set of advice for businesses about how to participate more actively in the development of the software they use, but from the questions he asks, it seems to me the scope of this is either beyond (eg Should I use one technology or another?) or totally outside (eg "Should I use open source or proprietary?") NZ GOAL-SE.

Of course, that doesn't stop anyone interested in helping develop such a resource from discussing or even organising it using this thread. Anyone who isn't interested doesn't need to comment or even to follow this discussion. This is one of the benefits of Loomio over having this whole consultation over an email list 🙂

Cam Findlay likes this.

Legal template for contracting

Started by **Edward Abraham** a month ago · Public

We develop software for government agencies. What I want from this policy is a succinct few clauses that can be put into contracts. Often contracting is a hurried and delicate process, and a well worded official template really helps. In the draft there is an example licence, but the policy should also include a legal template that covers the main issues.

ACTIVITY

RB **Richard Best**

As a lawyer who deals with these issues, I think this is a good idea.

Edward Abraham likes this.

Finlay Thompson

The advantage of a template is that it facilitates good contracting, and simplifies negotiating.

For example, the the GMC Form 2 SERVICES contract template is very useful for entering into small service contracts. Also, it includes in Schedule 2 a simple intellectual property section (Section 12).

Ideally we would have a Licence template with a small number (one digit) of options. The template should be supported with clear guidelines on selecting the option. Then, it could be referenced easily, and everyone would know where they stood.

Edward Abraham

For reference, the IP section from the GMC Form 2 is below:

> ## 12. Intellectual Property Rights
> Ownership of Intellectual Property Rights
> 12.1 Pre-existing Intellectual Property Rights remain the property of their current owner.
> 12.2 New Intellectual Property Rights in the Deliverables become the property of the Buyer when they are created.
> 12.3 The Supplier grants to the Buyer (as The Crown) a perpetual, non-exclusive, worldwide and royalty-free licence to use, for any purpose, all Intellectual Property Rights in the Deliverables that are not owned by the Buyer. This licence includes the right to use, copy, modify and distribute the Deliverables.
>
> Supplier indemnity
> 12.4 The Supplier warrants that it is legally entitled to do the things stated in clause 12.3 with the Intellectual Property Rights in the Deliverables.
> 12.5 The Supplier warrants that Pre-existing and New Intellectual Property Rights provided by the Supplier and incorporated in the Services and Deliverables do not infringe the Intellectual Property Rights of any third party.
> 12.6 The Supplier indemnifies the Buyer (as The Crown) in respect of any expenses, damage or liability incurred by the Buyer or The Crown in connection with any third party claim that the delivery of the Services or Deliverables to the Buyer or the Buyer's or The Crown's use of them, infringes a third party's rights. This indemnity is not subject to any limitation or cap on liability that may be stated elsewhere in this Contract.

Vic

Hi Ed, I am completely new to this whole consultation process and may be missing the mark here - spoken with my NZRise hat on. Since we worked with both the Government Legal Network (via DIA/GCIO's office) after the disastrous IP clauses issued by DIA late in 2015, and with the work done on the MBIE Procurement Reference Group to specifically focus adoption of those clauses you have posted on this thread I have had no NZRise members complain about Govt Procurement contracts attempting to deviate. To provide context there had been 1-2 contracts per quarter prior.

Equally the Guidance notes for NZGoal also provide strong IP treatment guidelines, including that the supplier can commercialise their own IP.
https://www.ict.govt.nz/guidance-and-resources/open-government/new-zealand-government-open-access-and-licensing-nzgoal-framework/nzgoal-guidance-notes/nzgoal-guidance-note-3/#_ftn2

Happy to assist with refining wording or clarifying specifically to reflect Open Source vs other IP license models if that is where you are heading with this. Vic.

Edward Abraham

Thanks for pointing this out @vic. It seems like the place to give example clauses for open source licensing, that can be slotted into the GMC. Hopefully @camfindlay1 can take you up on the offer to refine the wording.

Note that I also started a related thread about the ICT procurement policy, which would also need to be modified if open source licensing was to be promoted within government, so that there is consistency across these documents.

Strypey

Modify the ICT procurement policy

Started by **Edward Abraham** a month ago · Public

This policy should be closely linked to the IP section of the ICT procurement policy, see Gudeilines for the treatment of intellectual property in ICT contracts. This document gives three options for the treatment of IP:

> Three options are recommended for the treatment of IPR in ICT contracts:
> - The Customer Agency owns all new IP in the deliverables, with no licence back to the Supplier.
> - The Customer Agency owns all new IP in the deliverables, with a licence back to the Supplier for its commercial exploitation.
> - The Supplier owns all new IP in the deliverables, and provides a licence to the Customer Agency and other State Services agencies for any purpose other than commercial exploitation.

We have successfully used this policy in the last year to help us retain IP in software we developed for a government client. The decision tree in the document and templated contract clauses were extremely useful to help us discuss this with a client in a straightfoward way.

At least one of these options should recommend open sourcing. Ideally there is strong commitment from government (open by default) and this is the default option. It would be a shame to have a great open source licence but no way to get to it be following these IP guidelines.

ACTIVITY

Edward Abraham

Note that the ICT procurement policy is no longer online (except as a PDF). In particular this link is dead. Is the policy still active?

Strypey

This wording is hard to respond to, because it fudges together a number of totally separate areas of law under the misleading label "intellectual property". This catch-all phrase is presumed to cover copyright, patents, trademarks, and other areas of law, which each have their own issues, and need separate policy statements to address them.

If we presume that by "IP" the policy means trademarks, the wording is fine, as there is no way to libre license trademarks. For an open source project, I recommend we learn from what happened to the Koha project, and make sure any relevant trademarks are owned by the commissioning agency (or a suitable vendor-neutral stewardship body), under one of the first two options on that list.

If we presume that by "IP" they mean copyright (in either the source code or in interface artwork/ documentation), all three of these options assume a proprietary, exclusive approach, and we need to start again with fresh wording for open source projects.

- the first option is not an option, because any free code license delivers rights back to the supplier
- the second option is probably the ideal. Any free code license allows commercial use by the supplier
- the third option is also not an option, because for the reason given above, even if the supplier retains the copyright, it can't limit commercial use by the commissioning agency or anyone else

Dave Lane likes this.

Cam Findlay

See paragraph 11 in NZGOAL-SE draft. It seems the IPR in ICT contract guide from SSC has an out clause should a government agency want to retain rights in order to open source the work. This is on page 9 of the "Guidelines for Treatment of Intellectual Property Rights in ICT Contracts" at https://www.ict.govt.nz/assets/Uploads/Documents/ipr-guidelines-2008.pdf

I think therefore that NZGOAL-SE can be used alongside this other piece of guidance without requiring rework in the IPR guide.

Cam Findlay

Chiming in post consultation, a quick transparency statement. I'm going to be involved in helping draft the NZGOAL-SE revision, my opinion here is my own and I want to try and gain good understanding about how these two policies interact here to help guide my decisions through the revision process. On to some thoughts...

In terms of the IPR in ICT contract guide mentioned, following the decision tree in IPR in ICT, page 13, it mentions "*The contracted deliverables mainly use pre-existing IP (**which is already owned by the supplier**)*". In terms of taking an existing FOSS project they supplier doesn't "own" it as such and is reusing under FOSS license terms. Therefore it would seem to point the policy user towards the NO and "copyright owned by agency" end of the spectrum. This end is required to run it through the NZGOAL-SE review and release process.

If we think about what we are trying to do, end-to-end rather than siloed per policy I think a compatible and inter-linked chain leading to the desired outcome can be achieved, that of getting code produced through some procurement/commission, released under FOSS licenses for wide reuse.

Open sourcing something is not commercialising directly (to me this negates the "government is not in the business of commercialising software" argument for not holding the copyright). It's making a taxpayer funded, public good available as a base to innovate on top of. It allows parties in and outside of government to make use of and yes, commercialise as long as they follow the license (and this includes both permissive and copyleft).

My thoughts on the 3 positions (though this would appear to only be relevant in the case of wholly new software where the full copyright is vested in one party):

1. Customer Agency owns the new IP and decides whether to or who may commercialise
Copyright owned by agency they can then use the review and release process as per NZGOAL-SE.

2. Customer Agency owns the IP but licenses the Supplier to use and commercialise
Agency could release under open source license terms, if the intention is for the supplier to commercialise through incorporation into closed source applications or derivatives, a permissive license could be used publicly or granted to the supplier specifically and then publicly released under any license MIT/GPL etc for general public reuse.

3. Supplier owns the new IP but provides licenses to the Customer Agency and all other State Services agencies
To keep things simple, suppliers could use NZGOAL-SE to provide a FOSS license back to Government as a default. They could use a permissive license if they have discussed reuse in closed source derivatives. Further, the govt can then sublicense a public release under any other license (MIT/GPL) if they wish. Alternatively, the supplier could release under GPL for govt reuse, as they are the copyright holders they can then use a non-GPL'd copy of code to commercialise. They could release on behalf on a govt repo or on supplier repo.

Edward Abraham

> If we think about what we are trying to do, end-to-end rather than siloed per policy I think a compatible and inter-linked chain leading to the desired outcome can be achieved

Excellent! Great you are thinking about how NZGOAL-SE fits in as part of a broader framework.

Cam Findlay

From reading over the guidance on IPR in ICT contract in NZGOAL-SE draft, analysing the IRP in ICT contract guide, my thoughts are (with respect to my previous comment on this), that the current wording in para. 9-12 and 31-33 of NZGOAL-SE indeed align the IPR guidance and NZGOAL-SE as already compatible. We may be able to consider guidance notes on some practicalities on how to use these two policies together however, this is not to do with licensing (as NZGOAL-SE assumes you have already sorted out your copyright-related rights before moving further though the decision tree).

In light of this, we will be leaving this section as in the original draft. No changes to be added to the GitHub revision. 👍

Consider removing section 22 & 23

Started by **Cam Findlay** a month ago · Public

There was good debate and a lot of +1's for this in another thread. I'm raising a new thread here around this to close out this dialogue making use of the Loomio polling to capture the consensus around this.

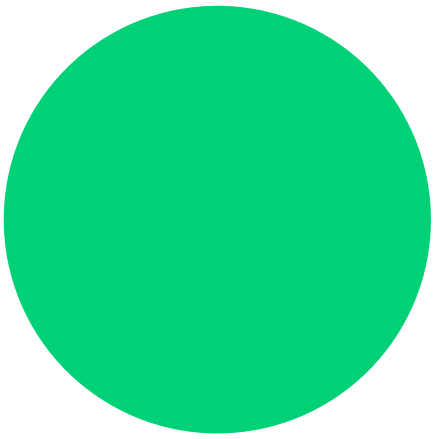
PREVIOUS PROPOSALS

Consider the removal of sections 22 and 23 ("stifling effect" of share-a-like)

Started by **Cam Findlay** · Closed 24 days ago

Please vote here to show you support for the consideration of removal of sections 22 and 23 from the NZGOAL-SE draft.

POSITIONS



7 Agree

0 Abstain

0 Disagree

0 Block

19% of members have stated their position (7/37)

Strypey agreed: I agree with the comments by Finlay, Brent, and Dave

Finlay Thompson agreed: These clauses are unnecessary.

Dave Lane agreed:
Those sections have an implicit "business perspective" rather than a "citizen perspective"... I think they're rather unnecessary.

Cam Findlay agreed:
To replace this section it would be good to explain (perhaps with case studies) some of the common misconceptions about share-a-like FOSS licenses like GPL.

Brent Wood agreed:

The so-called "permisssive" licences can be termed "stifling" as reuse & further development can be locked up in proprietary software & licences..
A more balanced description of the differences between GPL & BSD style licences should be provided.

Rob Elshire agreed: These sections are not necessary.

T. Charles Yun agreed.

Collapse

ACTIVITY

Cam Findlay started a proposal: Consider the removal of sections 22 and 23 ("stifling effect" of share-a-like) a month ago

Finlay Thompson agreed: These clauses are unnecessary.

a month ago

Brent Wood agreed:
The so-called "permisssive" licences can be termed "stifling" as reuse & further development can be locked up in proprietary software & licences..
A more balanced description of the differences between GPL & BSD style licences should be provided.

a month ago

Dave Lane agreed:
Those sections have an implicit "business perspective" rather than a "citizen perspective"... I think they're rather unnecessary.

a month ago

Rob Elshire agreed: These sections are not necessary.

a month ago

T. Charles Yun agreed a month ago

a month ago

Strypey agreed 24 days ago

24 days ago

Strypey agreed: I agree with the comments by Finlay, Brent, and Dave

24 days ago

Cam Findlay agreed:
To replace this section it would be good to explain (perhaps with case studies) some of the common misconceptions about share-a-like FOSS licenses like GPL.

24 days ago

Cam Findlay

Further, I note that in a publication by US DHS they cite from interviews with Government ICT staff that often there was misunderstanding of GPL. See https://www.dhs.gov/publication/host-co

Note: keep in mind the copyright aspect of govt work in the US is very different from our own here in NZ, all Govt created code is under *public domain*. It's often globally released outside US jurisdiction through CC0 which is not supported in the NZ context. NZ has Crown copyright (bundle of rights) when a govt staffer produces something like code (hence requiring licensing guidance).

Strypey likes this.

The proposal has closed: Consider the removal of sections 22 and 23 ("stifling effect" of share-a-like) 24 days ago

Strypey

@camfindlay1

> To replace this section it would be good to explain (perhaps with case studies) some of the common misconceptions about share-a-like FOSS licenses like GPL.

At the risk of drifting off-topic, one of the ongoing activities of CreativeCommons Aotearoa/NZ has been to write up case studies of kiwis using CC licenses, thus building up an archive of such case studies on our website. When 'A Quiet Revolution (CC-BY)', the recent book about CC in Aotearoa, was being put together, much of the material for it was already available in the case studies archive.

There have been a number of kiwi free code releases over the years, both inside and outside government, as well as a plethora of examples of kiwis re-using and contributing back to existing open source projects. Would it be useful to write up a set of case studies on some of these as a record of practice? For reference both by people working on NZ GOAL-SE itself, and by developers and agency decision-makers trying to follow the advice in it once it's released. With CC ANZ, this was made possible in part by the funding of a part-time paid position for the Project Lead. Do you think it may be possible to get funding for a similar paid position to help collate case studies about kiwi free code releases and open source practice?

Dave Lane likes this.

Cam Findlay

See https://github.com/opendatanz/nzgoal-se/pull/3 - proposal to remove.

OSS or FOSS?

Started by **Cam Findlay** 19 days ago · Public

When discussing licensing I usually use the acronym "FOSS" to get across the "freedom of reuse, modification" etc as well as being able to see the source code (that's the open source bit).

I usually avoid using use FLOSS (Free/Libre & open source software) because

- a) sounds like something from a dentist
- b) the "libre" bit I feel is only needed if we don't communicate well that "free" in free & open source software refers to "freedom". It's an education thing I think we're all responsible to share and make others aware of.
- c) one less acronym to remember is a good thing

I'd like to see FOSS used in place of simply OSS in NZGOAL-SE.

ACTIVITY

Strypey

I also think that just using 'open source' risks missing the point of free code software. For those who haven't read it, I highly recommend at least skimming the GNU.org essay on the subject. I agree with Dave that this could be dealt with by mentioning the Four Freedoms in the preamble, and maybe referencing the GNU.org essay.

Cam Findlay

See https://github.com/opendatanz/nzgoal-se/pull/7/files?short_path=d87f721#diff-d87f7210b76c2230ddcaec8658fa8be0 for the diff - I've included some info about the interchangeable language, the meaning of "Free" as in "Freedom" and indicated the particular language we'd use in nzgoal-se.

I'd welcome a fork and pull request if you have alternative wording for this, though I can't guarantee a merge, I'm keen to see how it might look through others eyes keeping in mind the intended audience - Government people - to whom this might be a whole new thing (though FOSS has been around for decades 😉).

Dave Lane

Just been reading a huge thread on the internat'l (but US-centric) "Foundations" list (you have to join to see the archives: https://lists.freedesktop.org/mailman/listinfo/foundations) re-litigating the "OSS vs Free Software" terminology discussion. Simon Phipps, one of the people behind the Open Source Initiative (OSI - http://opensource.org - they hold the trademark and vet licenses and whether or not they can legitimately claim to be "open source") said this, which I think is relevant (and interesting to see that their new initiative tries to sidestep the problem altogether):

"The fiduciary umbrella we've started for Europe (you'll remember I was asking about it last year) is called Public Software CIC, https://publicsoftware.eu/ and we picked the name specifically to avoid the terminology conflict and identity issue under discission, per https://publicsoftware.eu/about/why-public-software/

However, we also emphasise that the formula has to be licenses AND collaboration, since without the four freedoms guaranteed through an OSI-approved license, collaboration is probably illegal. One day that will matter a great deal to many of the innovators using open source today, when some amoral corporate behemoth decides to sue them for billions of dollars for doing what everyone knows was OK but which they struggle to prove was actually so.

In passing the baton to a new generation, it's crucially important we ensure their vision builds on rather than replaces that of the generation before them. We were fixated on licenses, but it was with good reason."

Dave Lane

Also, (another quote from the aforementioned Foundations thread) I find Bradley M Kuhn's (of the Software Freedom Conservancy and on the Free Software Foundation board) point resonates (it references the quote I posted above):

"I agree with Simon on this point. The fixation on licenses was *because* we
wanted to ensure the four key software freedoms for everyone.

I also agree with something Simon didn't say but he's hinting at: I think
there is a serious failure on the part of the previous generation (which
includes most of us) to explain that there were real, fundamental, moral
principles involved.

Does anyone have specific ideas of how we can communicate to the next
generation that there are moral principles and fundamental rights -- not just
"better code" -- at stake? I must admit that "public software" as a term
doesn't seem to get that across to me."

To me this is the burden that this NZGOAL-SE has to take on board - if we're recommending that taxpayer funded software be made available for the citizens, we need to explain our thinking as to *why* otherwise, the "spirit of the recommendation" is obscured, making it much easier for the clear commercial incentives to "obey the letter, but deny the spirit" to prevail.

T. Charles Yun likes this.

Don Christie

Hi

Interesting and thanks Dave. Simon Phipps (former Open Source and Java evangelist at Sun Microsystems) is an excellent source and thinker. He "gets" free software as well as understanding the thinking behind Open Source.

I really like that Public Software page and the thinking about needing to be explicit about allowing collaboration.

Cheers
Don

Grant Paton-Simpson

Open software

(vs FOSS or OSS)

Open means more than access to viewing source. And it's shorter.

Dave Lane in reply to **Grant Paton-Simpson**

My concern with that, Grant, is that it's even more vague than open source... and has already been co-opted by many (e.g. NZ Labour) who don't really understand what any of free, open source, or libre mean...

Grant Paton-Simpson

Heh - probably right.

Adaptions (para. 19) should cover using same license as adapted project.

Started by **Cam Findlay** 19 days ago · Public

This section could include explicitly that if adaptions are made to existing open source software that it should be licensed under the same license as the original code base or one in keeping with that particular open source ecosystems community norms recommended in NZGOAL-SE. I'm not sure if this was explicitly mentioned in other sections of NZGOAL-SE draft?

For example, a government agency releasing a SilverStripe CMS module could use BSD or MIT (SilverStripe CMS is originally under BSD). MIT's "permissiveness" fits in with the original projects BSD origin if the agency wanted to keep tight to NZGOAL-SE. Same deal if adapting something like Moodle (GPL v3) the Moodle community would expect GPL adaptions back.

ACTIVITY

Cam Findlay

This could actually be a very early question in the decision tree "Are you adapting or releasing code that interacts with an existing FOSS project?" - YES -> "Release under the same license as the existing projects code base" END OF TREE.

Strypey likes this.

Strypey in reply to **Cam Findlay**

It would be really interesting to get some metrics on how many projects come into this category (modifying existing project), and how often entirely new software is (or could be) released by NZ govt as open source projects.

JR **Jason Ryan**

My experience is that government almost always adapts or uses an exisiting project, and even when they do build something it is using a framework or platform that has a FOSS license. It makes sense both practically and philosophically to use the project or framework's existing license. It is also the Right Thing to do...

Liked by Dave Lane, Cam Findlay and Strypey.

Dave Lane in reply to **Jason Ryan**

Yes, that's my experience, as well. See, for example, CKAN, Drupal, SilverStripe, etc.

Strypey

If the answer to the question "is the new code modifying or building on top of an existing open source project" is always "yes", is "use the same license as the upstream package" the only piece of advice NZ GOAL-SE actually needs to give? This advice would make the document *very* simple to understand and follow, and would cover both:
1) situations where there is a legal obligation to share the modifications under the same license. This is case whenever copyleft code is modified and distributed (keeping in mind the distribution conditions that trigger the copyleft condition in each license)
AND
2) situations where the copyright owner has the freedom to choose a license, but using the same license governing the upstream package reduces complications for the its maintainers, as well as re-distributors and re-users. This is the case when:

- non-copyleft code is modified
- copyleft code is modified and distributed in ways that don't trigger the copyleft clause (eg modified GPL software used on a server without distributing binaries or source code)
- a component is built from scratch to work in combination with an existing free code package (eg a module or plug-in)

If a government department was proposing to release code created from scratch, for in-house use, they are effectively creating an entirely new open source project, which would require advice on an awful lot more than licensing. If this is unlikely to happen in practice, because government is not in the business of software development, does NZ GOAL-SE need the much more complicated decision tree about choosing a license for new software at all?

Cam Findlay

See https://github.com/opendatanz/nzgoal-se/pull/8/files?w=1