It's not just code: managing an open source project

Tracy Teal

2023-02-25

Table of contents

Overview

This workshop is about Managing an Open Source Project.

All content is currently in very draft form!

We'll cover these topics.

- Identifying the goals for your project
- Setting up your project
  - Managing expectations
  - License
  - Contributing guidelines
  - Code of Conduct
- Documentation
- Governance and decision making
- Commununity interactions
  - Responding to issues and PRs
  - Managing conflict
- Project management
- Maintenance
- Funding

1 Introduction

Rough Draft

What can make or break an open source project? Good code? Need for the project? Yes, all of these things are important! But there's another elemetn important to the success of an open source project and that's how it's managed. We tend to think of the work 'managed' as 'having control over something', but management of an open project is more about sharing and distributing control in a way that still allows decisions to be made, existing people to be supported, new people welcomed and having a structure that external people can understand to know how to support it best (including funding it!). This is even more important as projects are moving away from the BDFL model to governance structures and collaborative decision making. This is an important transition in the sustainability and equitable participation in open source. And open source itself continues to be foundational for the technology space for all the reasons Nadia shares in her Roads to Bridges report.

If this sounds hard though, it is! What's even more hard abotu it is that most of the people in these open source 'management' positions started their project because they wanted to code. They usually haven't had the opportunity to get experience or training in facilition, finances, and other things. Writing code not mean that someone necessarily signed up to run a business! But yet here we are. So, where do we go from here. One, this is something that can be learned; two, there are resources and practices that already exist that we can use; and three, we don't have to learn it alone!

While all of this is true, it's also true that each project is different, with different people, stages and ecosystems. There is not one solution or approach to anything (alas).

This workshop is a set of resources, discussion and questions for you to think about to help you figure out what is right for your project, and you, right now.

Most importantly, know that you're not alone. You might feel like you have no one to talk to about the challenges that you're facing, that you're doing a terrible job, that you're doing a great job (often both in the span of a day). You're doing fine! (Not to be confused with 'Everything is fine.') Let's get started learning together!

2 Project Goals

What kind of project are you? Like any good story we start with more questions than answers. The frist thing to consider is what kind of project are you and what kind of project do you want to be. So, let's first consider these questions.

- How many people use your project?
- What type of people use your project?
- What percent of people in a field use the project?
- How many contributors do you have?
- What type of contributors?
- How long has the project been around?
- How is the project, or the people working on the project, funded?
- Who are all the stakeholders?

.... more questions

3 Project Setup

Setting up your project is about setting expectations - about what you expect from people and what people can expect from the project.

People have a lot of different ideas of what open source means, if they're new or have been around awhile, what other communities they've worked with. They'll bring all of these things with them to your project. To ensure you and they have the best experinece with your project, it's good to be clear about expecationa, what they can expect from you and you can expect from them. This means you need to write it down and put it somewhere expected and ifnablae. It does not mean everyone is going to read it who comes to your project, but it's something that is theer for onbaording and something you can point to in replies. While we're talking about open source, and that means certain things, there are many different modes of 'open contribution'.

Now, when do you need to write these things down? Sonner than you think you do, but not too soon. There are some things you need to write down write away, and others that can come later or change over time. We'll explore this more in different sections too.

There are few things to set up from the start! These are:

- License
- Code of Conduct
- Contributing file - often CONTRIBUTING.md

From the beginning you'll need to be thinking about communiter interactions and how you want to handle issues and PRs.

3.0.1 License

You'll need to pick a license. There are several types of open source licenses. Importantly there are a clear set of approved OS licenses, so you should pick one of those. These are some of the common ones, with a very general description of each.

You should think about your project goals when selecting a license, and also what capacity you have for enforcement. If you pick a restrictive license, is that something you'll be able to follow up on, why or why not might that be important. What might you want to do with the project in the future.

A license is harder to change in the future, you usually need to get the permission of all contributors to change it in the future. So, it's worth taking some time to think this through at the beginning.

If you've already selected a license and find that you really need to change it, here's some ways you might go about it.

3.0.2 Contributing file

Another thing to think about is your contributing guide.

- How do people contribute logistically?
- What should people's expectations be for a realistic timeline for responses or reviews?

3.0.3 Code of Conduct

From the get-go you want to establish some fundamental community norms. We do that now through a Code of Conduct. These are now widely accepted for open source projects. There are some hold outs, but to be a modern open source project, you don't want to be one of them. The CoC signals to people that it's something the proejct cares about and that there are ways to share concerns.

Conveniently GitHub now makes this easy. While you used to have to draft your own CoC, GitHub has done good work to create a template Code of Conduct that includes descriptions of acceptable and non-acceptable behavior and enforcement guidelines. So, when you set up your repo, or if you need to add one, you can use [CoC link].

Now let's take a look at the CoC. There's the 'behaviors' part and the 'enforcement' part. Both are important! The standard behaviors list should be fine for most projects, but if you're a community with certain expectations that go beyond what's in this list, you can certainly amend.

Enforcement is the key part of a CoC that people talk about less. Having expectations is one thing, enforcing them is another. You need clarity on how people can report what they see as violations, and what can be expected when they do.

So, the key thing that you need to fill in, is that email address people can use to report. You should also share who receives that email address, so people know who they're sending it to. Ideally it should be two people, but while that's not always an option, it can also be good to have another point of contact, in case the issue someone has is with the person receiving that email.

Very important! When someone reports something, it is not up to them to be deciding if someone 'gets in trouble'. You want to take that weight from them. You want them to share

and then it's up to the people on your project enforcing the CoC on what steps to take. Being on the other side of a CoC email address is non-trivial, and it takes work and practice. [This is a good resource for learing more] and if you have a large project, I'd recommend a course like [Otter tech].

Other things you can do is share things like transparency reports. Again, go back to your project goals to decide if something like this is important for your project.

3.0.4 Summary

These are all your starting places, as your project gets foing they may change. Your license is hard to change and your CoC shouldn't change too much, but these are the foundation you'll build on.

4 Documentation

Another big piece for you and others working with the project is documentation. Here you'll want to review that 'what type of project are you'. If this is a project for astronomers only, you can assume astronmy knowledge when you write your docs. If it's software that started in astronomh, but now you're broadenign who it works for the you' 'l need to start at a different place in your docs and may even have to use a different format.

The minimum set of documentation here is something like a pkgdown page. [Link to pkgdown]

The next set of things are 'guides', documentation on how people can do the common set of things you would want them to do and 'reference', how different functions work.Depending on your code and documentation infrastructure, reference could come from your doc strings.

Closely related to guides are 'examples'. What are some examples of what you would want people to use.

If you're looking to stand up a set of documentation, and haven't started yet, Quarto could be an option for your docs. Here's a rough template [link to Quarto docs template that I haven't created yet].

A note, documentation can be hard! It's its own skillset and set of work to do. So, you will need to spend time and focus on it. The great thing about it being a skillset though, is it's something you can practice and learn! A great way to start is to find and read through documentation you like. What do you like about it? What might you emulate in your project?

You also want to encourage contributions to your docs! As mentioned, documentation is a skill, and even harder when you're not intimiately familiar with a project. So, "please write documentation" is not a particularly useful set of directions. People are great at offering feedback on what's already there though, so you can encourage people to put in PRs to fix things they see as wrong or file issues. This really helps vet your docs, and will help other people too when things are correct. Often we suggest documentation as a good first contribution. That too can be true, but that person will need a pretty clear set of guidance on what to contribute to docs, just like they would to code.

5 Governance and decision making

Governance is decision making. You set up governance structures, so you can make decisions for the project. Here again you need to think about project goals.

I'll start by being a little controversial here. You don't always need a governance structure. If it's just you working on the project and you're making all the decisions, and it's at an early lifecycle stage, that's probably fine. That's maybe less fine as the project beomces more central or other factors though. And here's where I back off from that initial controversial claim. We can rephrase it as a single person making decisions is its own form of governance. We just don't often think about it that way.

So, let's talk about a few differnt models.

5.0.1 Single maintainer

So, it's just you. You're the primary contributor and docs writer and all the things. So, clearly you make all the decisions and everyone knows that, and it's fine, right. Alas, no, 1) some people might not know that and 2) you still need to decide and communicate how you make decisions. Think of future you here, will you understand how past you made decisions.

There's not necessarily an obvious file or section to write about a governance model. (look to see if something like governance.md is standard). However, you won't have much to write here, so you could include some text in your CONTRIBUTING.md, like

```
Decisions for the proejct will be made with (list or link to project goals) in mind and the r

If you''d like to propose a new idea for the project, please share it an issue for discussio
```

Again, this will be something you can point back to if, e.g. someone puts in a PR that's out of scope or will be hard to maintain.

5.0.2 Multi-person project

You're growing or started this project as a collaboration. The main thign you need to do is make some decisions abotu how you'll work together and make decisions. There's a lot of models for this.

Obviously more...

5.0.3 Multi-stakeholder project

How is a multi-stakeholder project different than a multi-person project? It's different because you are at the stage where you have multiple types of stakeholders - users and contributors who may have competing needs or interests. This is where governance really shines. You need to be clear about how decisions for the project are made that serve the community as a whole. Otherwise you can end up in situations where work gets done just where people can put effort, and that might not be the top set of priorities for the project as a whole.

This is definitely complicated! Here are some models and ideas.

6 Community interactions

Topics to include:

- Handling issues and PRs
- Conflict resolution

7 Project Management

Project management is its own skill, and it's not usually taught! But it's a superpower! If you are able to manage a project, and not necessarily do all the work yourself, your project capacity expands immensely.

8 Maintenance

Things to include:

- Lifecycle (also in project goals)
- Spring cleaning

9 Funding

Is this maybe the topic you've been waiting for? Well, here's the good news, all that you've done to set up your project has put you on a good path for attracting funding. Yes, as with everything else, there's still work to do, and skills you need, but one key is having a project structure that shows that you as a project have goals, the capacity to make decisions and a structure that supports the project and community.

Alas, funding won't come to you just because you're an important project. Queue xkxd comic of the internet infrastructure resting on a small stone maintained by one person in Nebraska. Now, here I am describing the world as it seems to be, rather than the world as I wish it was. We can have a whole different conversation about that, but let's start where we are.

Here of course, we again start with questions. This will help you decide where to look for funding and how to frame asking for it.

- What would funding allow you to do that you can't do now?
- How would you make decisions about how to use the month, should you get it?
- Is there a business model (this is a whole deal of course) that can be made around the project?
- How much money would make a difference?
- How will people give you money?
- Do you have a bank account for the project, where will you get and store your money? (please not under a mattress)

When you're asking for funding, you can point to the information you have around how decisions are made, to let them know. Then you can say if I have X dollars, I would have more maintenance capacity, so I could make decisions for the project differently. Or if we had Y dollars, I could expand the scope of the project to do this new thing.

10 Summary

In summary, this book has no content whatsoever.

```
1 + 1
```

```
[1] 2
```

References