# For a more Transparent Governance of Open Source

(non-edited version of the manuscript CACM-22-06-4283.R1 accepted at Communications of the ACM
https://dl.acm.org/doi/10.1145/3570635 )

Our modern society runs on software. Most of this software is, or heavily depends on, open source code maintained by a community of, often, unpaid volunteers. Indeed, Free and Open Source Software (FOSS) constitutes the digital infrastructure of our society [Eghbal16].

While this openness is positive, it comes with a price. FOSS, as other types of public goods, suffers from a participation inequality problem: everybody uses FOSS, but few contribute back and hugely critical projects end up being maintained by very few committed individuals.

The long-term sustainability of FOSS is a complex and multi-dimensional problem (technical, economical, social, political, etc.). We believe more transparency in how projects are governed would be a significant improvement to all such dimensions. And one that it is easy to implement.

The lack of key governance information deters potential contributors, as they may feel the onboarding process would be too time-consuming [STG19, SSF+19] or may fear there are hidden power relations in the project that could limit their impact. The same goes for end-users, which may decide among similar projects based on how healthy and transparent the community behind them is.

To address this, FOSS projects should be more transparent and explicitly publish how they are governed in an easy-to-find and easy-to-read file[1] acting as the single source-of-truth for the project. This file should, at least, cover aspects such as the project's: (1) contribution workflow, (2) decision process to accept new contributions or prioritize features, (3) timeline for making these decisions, or (4) steps to climb the ladder in the project internal organization.

We are not there yet, as our analysis data shows. But we hope the discussion and recommendations that follow will help turn the tide and motivate projects to move towards a more transparent governance so that we can then discuss what the best governance models are, depending on the project characteristics.

## How transparent is FOSS governance? Looking at the data

FOSS data, especially regarding projects hosted on GitHub, the most popular social coding platform, has been analyzed from many perspectives to learn how FOSS communities collaborate. But, so far, none of the works have focused on the evaluation of their transparency and governance dimension.

Therefore, to evaluate the transparency dimension, we conducted ourselves three preliminary different analyses. Each one narrows down the number of analyzed projects but widens the depth of the analysis.

We first queried the over 200 million repositories in GitHub for any mention of the word "governance" in their readme file. Only 21,114 (a tiny 0,01%) were a hit.

Next, we focused on four specific software development ecosystems to run our analysis on more homogeneous sets of projects, namely: NPM packages, R packages, Laravel packages and WordPress plugins. We gathered all repositories from 2017 to now, and searched for governance information. To broaden the search, we looked for specific governance files but also looked into contributing and code of conduct files that could include governance aspects.

We collected information from a total of 13,937 repositories. None of them included a `governance.md` file. And the presence of contributing and code of conduct files was also low.

**Table 1. Presence of specific files in GitHub repositories for NPM package, R package and WordPress plugin ecosystems since 2017.**

| Ecosystem | # Repos | File | | |
|---|---|---|---|---|
| | | Governance file | Contributing file | Code of conduct file |
| NPM package | 4,636 | 0% | 6,23% | 4,36% |
| R package | 1,826 | 0% | 16,10% | 15,12% |
| Laravel Package | 1,447 | 0% | 23,43% | 5,46% |
| WordPress plugin | 6,028 | 0% | 4,05% | 2,17% |

Both analyses above were fully automated. As such, we could have missed governance information scattered in other project sources or matched projects that in fact were using the term with a different semantics or where governance information was minimal.

Therefore, we performed a final, more in-depth, analysis of the top 25 starred GitHub software projects. We looked for key governance information (recall previous section) in contributing guidelines, code of conduct, readme and project metadata (exploring and following any links that may be provided).

[1] We and others call it `governance.md` but the community should agree on a concrete name.

Figure 1 shows the results[2]: 60% of the analyzed projects did not include any governance information while 32% partially discussed governance but only covering two or three aspects, not all of them. In particular, the most ignored dimension is the expected timespan to accept/review a contribution. Only 8% of the projects (i.e., React and Kubernetes) mentioned all governance dimensions. And only one project (i.e., Node) included a `governance.md` file. Finding the evidence for the rest forced us to read and navigate a number of locations and project resources.

Unfortunately, the current situation does not show a significant improvement regarding past analysis[3].

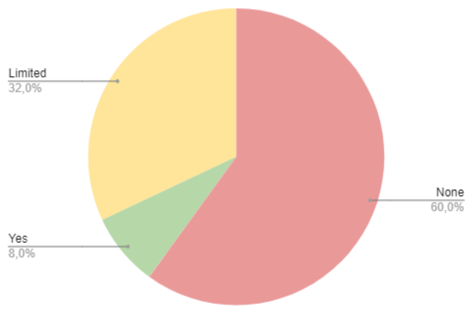

**Figure 1. Governance evidence in the top 25 software GitHub projects.**

## Towards a more transparent governance

Based on the above results, we all (project owners, contributors, researchers, users, etc.) can clearly do better. This section discusses a potential roadmap to improve transparency in FOSS by focusing on the following items:

- *Identify and learn from the best.* Transparency is not common, but this does not mean that there are no projects that do an outstanding job in explaining in detail their governance. Well-known projects such as Node, Drupal or Debian, to mention a few, can be used as inspiration of different approaches to governance. Ideally, we should find examples across different categories (size, language, topic, etc.).
- *Agree on a single place to keep the governance model*. If the governance rules are part of the implicit community "know-how", and scattered in many places (or too difficult to find), it is as good as if they were not existing. We propose to place all governance info in a `governance.md` file in the root folder of the project.
- *Be precise with the definition of the governance policies* by using a Domain-Specific Language (DSL) created for this purpose ([CC15] would be a possible starting point). This has the added benefit that then the governance description can be automatically processed, opening the door to interesting features such as the automation of some project decisions or a project search based on governance characteristics. The latter, for instance, could be useful for potential contributors only interested in contributing to projects that commit to quickly process pull requests. The lack of proper tools to define and manage governance is one of the reasons hampering its definition [Jansen20].
- *Offer predefined templates for the most common governance models*, described using the previous DSL and aligning them with initiatives like Minimum Viable Governance[4]. This way, new projects do not need to start from scratch. Instead, they just customize the template to their specific situation.
- *Collectively pushing for projects to add this information*. If there is no governance information, ask for it! This is a useful way to signal to projects that they must be more transparent. Transparency could also be set as a mandatory requirement imposed by foundations to any project applying to join.

All these initiatives will lead to a more transparent open source ecosystem, together with other existing transparency initiatives such as the code of conduct and the contributing guidelines files, also starting to timidly appear in projects as seen before. And it would open the door to start discussing the disclosure of further managerial aspects of the project (such as financial aspects and conflicts of interest of company contributors) that are mostly opaque nowadays.

## But what would be the best governance model?

If we agree on the importance of defining an explicit governance model for FOSS projects, the immediate follow-up question is whether there is an ideal governance model for FOSS.

We do not think so. The idiosyncrasy of FOSS projects  is so varied [Moz19] that there is no one-size-fits-all model. Even, so-called, "benevolent dictator for life" models are tolerated and seem to work well for some projects. It is also easy to find examples of meritocratic systems (where community members are promoted to some kind of steering committee based on their, mostly technical, contributions). Instead, large projects truly committed to an open participatory, or at least representative, democratic model are scarce[5].

While there is no ideal governance model, we believe there are a few general recommendations to consider when deciding it.

To begin with, dictatorships should be the exception and restricted to the beginning of the project. Otherwise, there is a high risk of the project being forked and the community splitting.

Instead, we suggest evolving towards one of the many more collective and participatory decision models [SEP13]. A notable example has been Python. After Guido van Rossum stepped down, Python has moved to a steering council model. These decision models include democratic ones, especially when you keep in mind there are hundreds of different democratic models [Gagnon18] to choose from. Indeed, we have observed that some projects react negatively to the concept of becoming more democratic as they have a narrow perspective which assumes that being democratic means a specific participatory model where everybody can vote, and all votes have the same value. But this is by no means a necessity: Instead, one could, for example, give to each person a number of votes depending on how valuable their contributions have been.

[2] Analysis done on May 13th 2022. Results available here

[3] Analysis in 2016 and 2018.

[4] https://github.com/github/MVG

[5] The Debian constitution would an example of a project close to being democratic.

We also suggest making sure that all types of profiles, not only technical ones, are included in the governance model. They are all important in their own way [CC22] and should have a saying. Again, this does not mean that a plain end-user should have an opinion on architectural decisions but it does mean the model should guarantee their participation on the aspects of the project relevant to them are heard. Having a voice increases perceived fairness [LKE90].

Finally, we would also recommend considering external factors in your decision. As an example, if your project is funded by external sources, you should think whether they have the right to influence the project. Similarly, if you aspire to join a certain foundation or attract certain types of contributors, the openness of your model could be a factor to entice them.

One way or the other, as a society, we have thousands of years of experience in testing many different models throughout our common history. While these past experiences are not directly applicable to the FOSS world, they are for sure valid inputs that can teach us how communities typically evolve and what risks are to be foreseen. As FOSS has had a much shorter history so far, let's learn from them to ensure healthier communities and the long-term sustainability of FOSS projects.

## References

[CC15] Javier Luis Cánovas Izquierdo, Jordi Cabot: Enabling the definition and enforcement of governance rules in open source systems. ICSE (2), 505-514. 2015

[CC22] Javier Luis Cánovas Izquierdo, Jordi Cabot: On the analysis of non-coding roles in open source development. Empirical Software Engineering, 27(1):18. 2022

[Eghbal16] Nadia Eghbal. Roads and Bridges. Ford Foundation. 2016

[Gagnon18] Jean-Paul Gagnon. 2,234 descriptions of democracy: An update to democracy's ontological pluralism. Democratic Theory. 2018

[Jansen20] Slinger Jansen: A focus area maturity model for software ecosystem governance. Information & Software Technology, 118. 2020

[LKE90] Lind, E. A., Kanfer, R., & Earley, P. C.. Voice, control, and procedural justice: Instrumental and noninstrumental concerns in fairness judgments. Journal of Personality and Social Psychology, 59(5), 952–959. 1990

[Moz19] Open Source Archetypes: A Framework for purposeful open source. Mozilla Foundation. 2019

[SSF+19] Dan Sholler, Igor Steinmacher, Denae Ford, Mara Averick, Mike Hoye, Greg Wilson. Ten simple rules for helping newcomers become contributors to open projects. PLoS Computational Biology, 15(9). 2019

[SEP13] Stanford Encyclopedia of Philosophy. Social choice theory. https://plato.stanford.edu/entries/social-choice/. First published Dec 18, 2013. Accessed August 2022

[STG19] Igor Steinmacher, Christoph Treude, Marco Aurélio Gerosa. Let me in: guidelines for the successful onboarding of newcomers to open source projects. IEEE Software, 36(4):41-49. 2019