



# Governance for Software Engineers

**Open Source Summit Europe  
September 2024 – Vienna, Austria**

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# Governance for Software Engineers




-  **Part I:** What is governance? – demystifying open source project governance
-  **Part II:** A Simple, practical, and proven approach to **writing and maintaining** project governance directly inspired from coding best practices





# Who am I?

## Tobie Langel

-  Jazz drummer → open source dev → consulting
-  UnlockOpen, open source strategy consulting firm
-  OpenJSF CPC Vice Chair & Board, Chair W3C  
Coremob CG, Facilitator AMP AC, etc.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# Part I – What is governance?\*

\* of an open source project or its foundation

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# Part I – What is governance?\*

1. Working definition for FOSS governance
2. Delegation of authority
3. Governance is bounded

\* of an open source project or its foundation

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



*“Governance is the **formalization** of **implicit norms** and **culture** in order to **scale collaboration**.”*

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



*Let's unpack this.*

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



*“Governance is the formalization of implicit norms and culture in order to **scale collaboration.**”*

**Scale collaboration** - governance needs are function of size of contributor pool and impact of project.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)





*“Governance is the formalization of **implicit norms and culture** in order to scale collaboration.”*

As a project grows, **implicit norms and culture** have to be uncovered and spelled out.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)






*“Governance is the **formalization** of implicit norms and culture in order to scale collaboration.”*

**Formalization** – spelling-out norms and culture isn’t enough. It has to fit within the existing system of **authority delegation**.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)

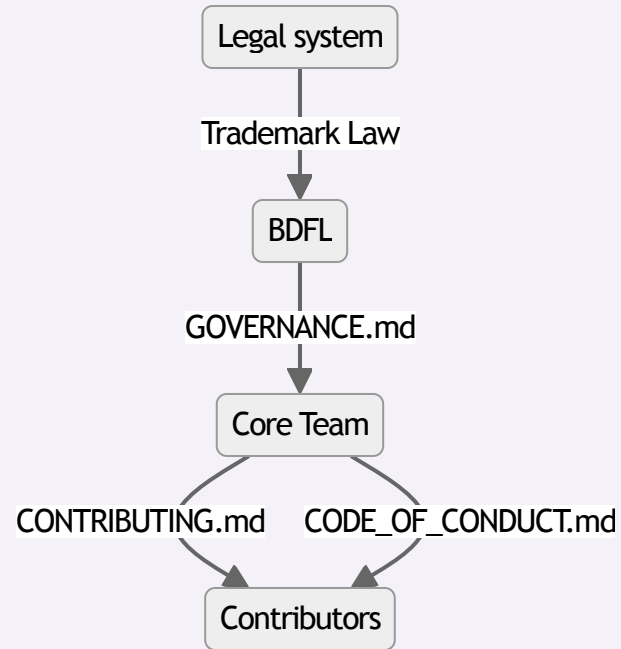


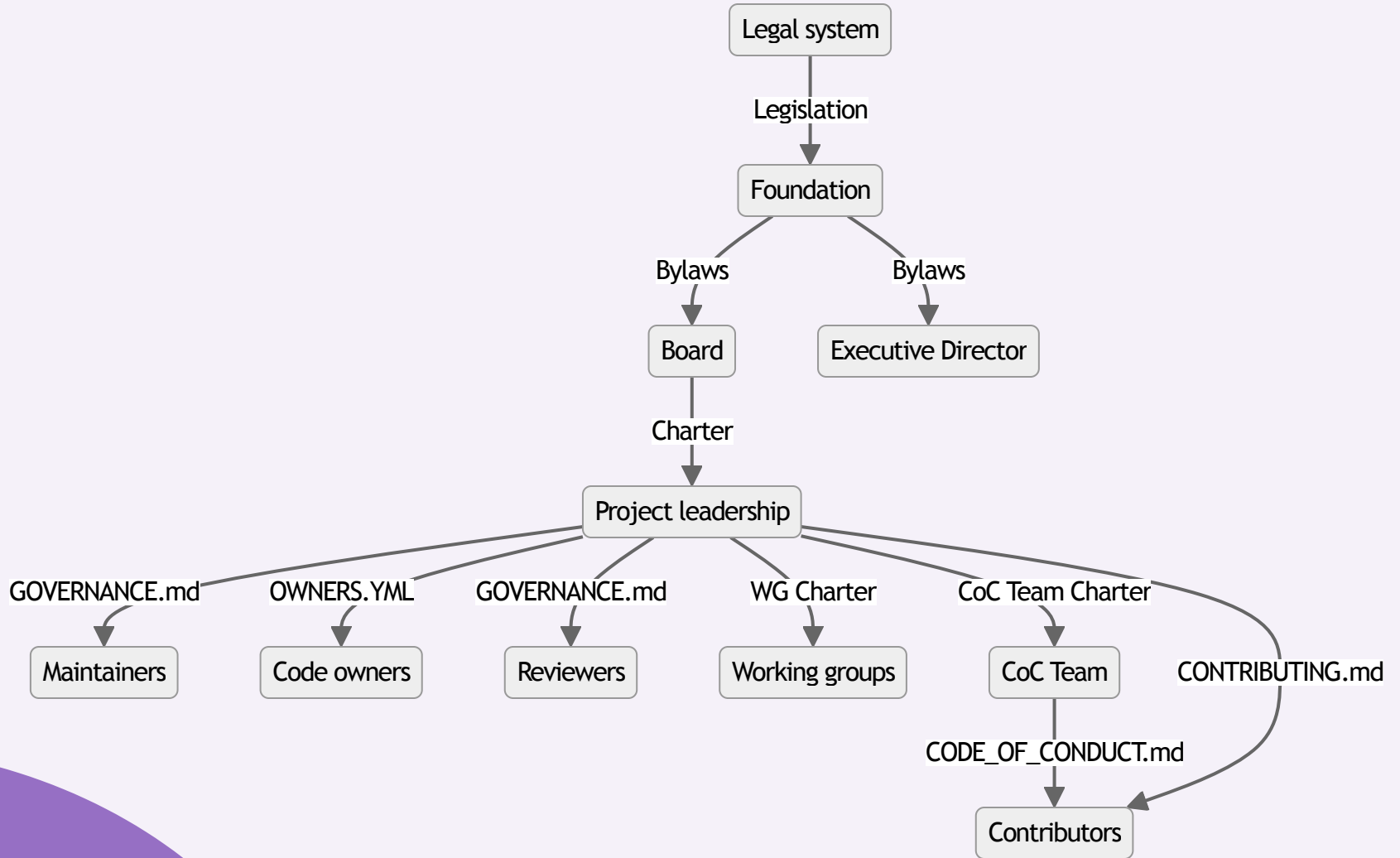
## 2. Delegation of authority

-  Starts with **legal framework** that recognizes certain rights and **delegates authority** accordingly.
-  Can be **explicit** (e.g. foundation) or **implicit** (e.g. copyright & trademarks owned by project creator).
-  Can be **simple** or **complex** (e.g. Open AI\*).

\* As we'll see, sometimes governance isn't everything; people might decide they no longer want to play ball and just leave. Whatever governance system you have in place becomes meaningless.

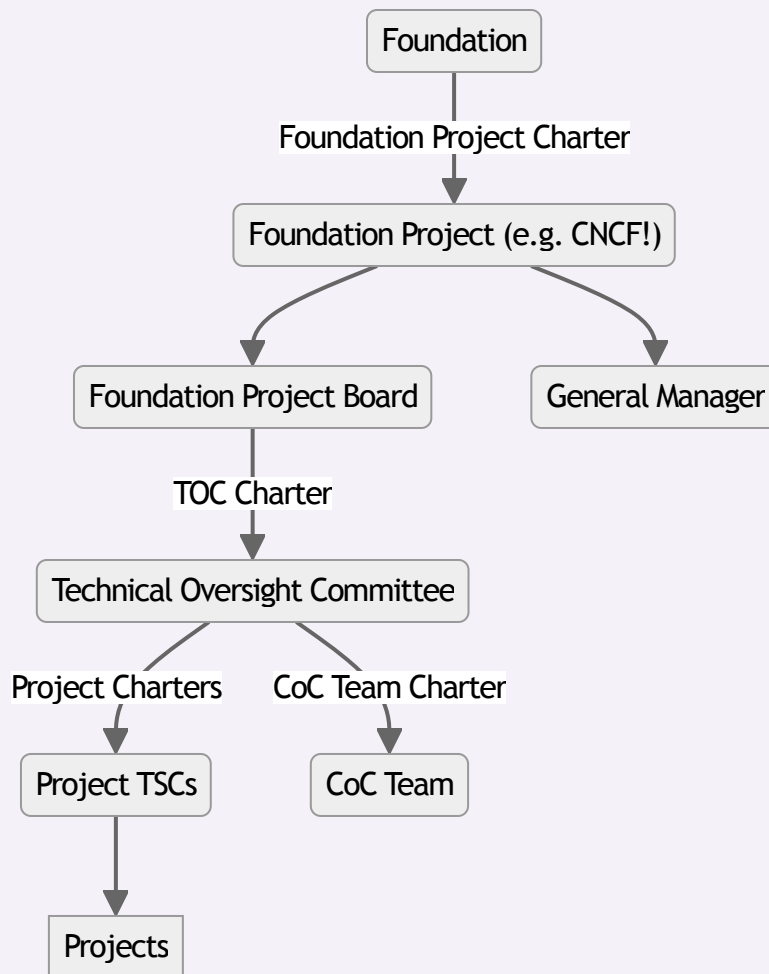
Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)





Large project in a dedicated foundation

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)

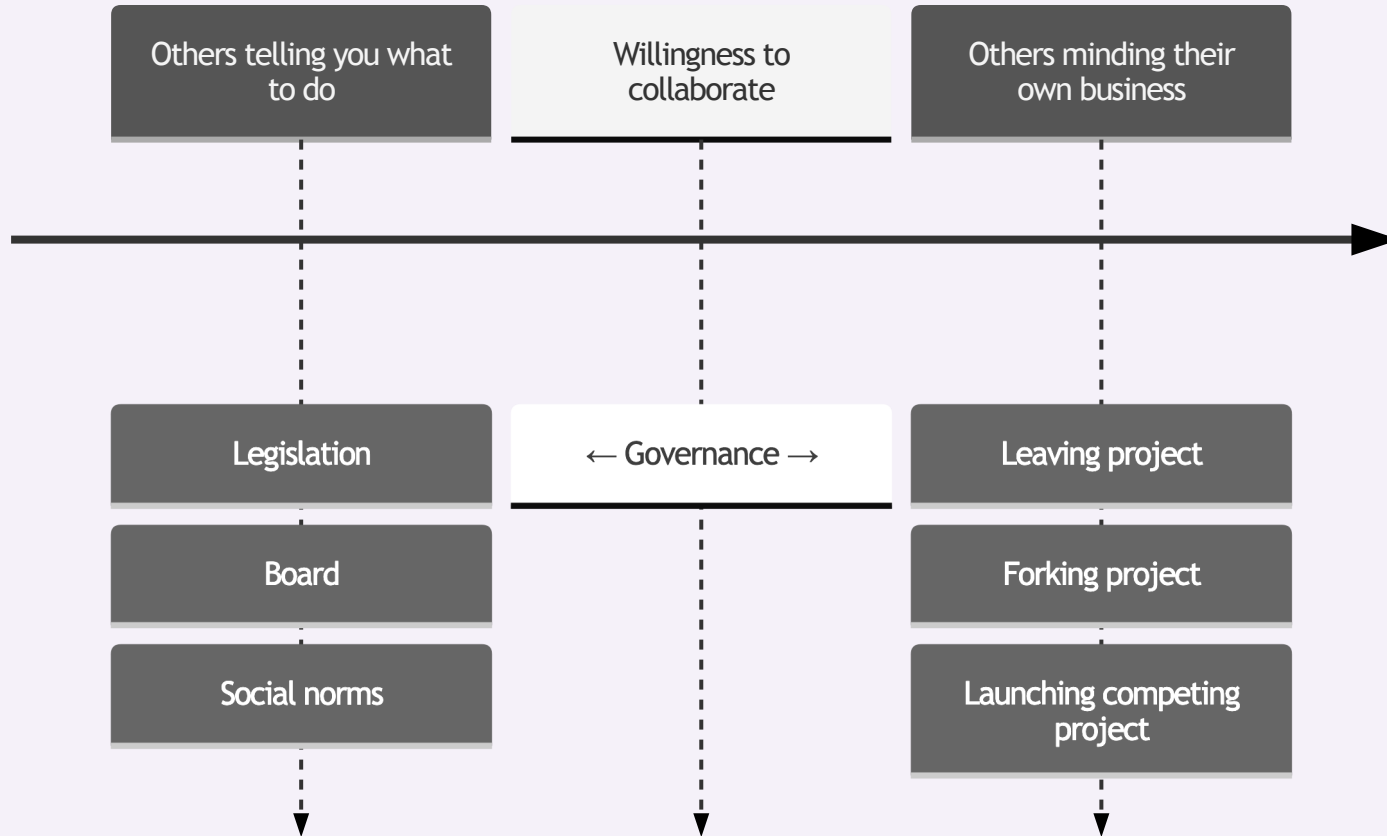


Multiple projects hosted by a *foundation project* (i.e. not a software project), itself part of a larger foundation.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



### 3. Governance is ← bounded →



Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



## 3. Governance is ← bounded →

- Governance exists as long as there's **willingness to collaborate**.
  - 🤝 You can think of governance as the organization of willing collaboration.
  - 👣 If you're unreasonable, people will leave.
- You remain **subject to the authority** of the entities that have power over you.
  - 🚫 You can't substitute yourself to existing authorities.
  - 💰 Power might be direct (e.g. board, legislation) or indirect (e.g. membership fees, trademarks, key person leaving, ...).



# Part I - Takeaways

- 🤝 Governance is the **organization of willing collaboration** – it's bounded by delegated authority and other's willingness to collaborate with you.
- ⚙️ Governance **fits within an existing system** – understand what parts of it you're formalizing and stick to those.
- ⚖️ Governance should be **proportional to project reach and complexity**. Excessive governance yields *bureaucracy*, write as little as possible.
- 🚫 **Avoid aspirational governance** – spell out **existing norms**; do not invent new ones.
- 🕒 **Convey intent** – turning norms and culture into rules and obligations is a lossy process. Say *what* you're trying to achieve and *why*.



# Updated working definition

*“Governance is the formalization of implicit norms and culture in order to scale **willing** collaboration.”*

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



**...or shorter**

*“Governance is the organization of willing collaboration.”*

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# Part II – Writing & Maintaining Governance

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)





*Governance is code that has to be interpreted by humans.*



*Governance is code that has to be interpreted by humans.*



*Everything you know about writing good quality code  
is even more relevant!*



# Part II – Writing & Maintaining Governance

1. Common issues
2. Best coding practices as a solution
3. Simple framework to get started
4. Additional tips



# 1. Common issues

Same issues in governance as in code:



Spaghetti code



Copypasta







High coupling & low cohesion



Lack of structure



## This leads to:





-  Maintenance nightmare
-  Governance that's hard to modify and update
-  Governance that's difficult to understand and follow
-  You start having two realities:  
What's on paper, and what's really going on





## 2. Best coding practices as a solution






Same coding principles apply to governance:

-  Good architecture
-  Low coupling and high cohesion
-  Separation of concerns
-  DRY, etc.








# 3. Simple framework to get started

Limit yourself to these 5 document types:

Type	What it's for	
 Charter	Authority delegation	<i>Who?</i>
 Policy	High-level goals	<i>What?</i>
 Process	Implementation of these goals	<i>How?</i>
 Guidelines	TL;DR for a specific purpose	
 Documentation	Everything else	








# 3. Simple framework to get started

Type	Programming equivalent
 Charter	Framework / Config file / <code>main()</code> ;
 Policy	API / header file / Type def / Interfaces
 Process	Actual implementation code!
 Guidelines	High-level API / Façade
 Documentation	... documentation !



# 3. Simple framework to get started

	Type	Owner
	Charter	Delegating authority
	Policy	Delegating authority
	Process	Implementors
	Guidelines	Depends
	Documentation	Depends

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# How this works in practice #1

1. You write a **policy** about specific goals you have (the *what*).  
👉 *Bonus points if you explain **why** those goals matter.*
2. You **charter** a group to implement that policy.
3. That group implements that policy through a **process** it controls.



# Example: Moderation policy for GitHub

1. You writes a **policy** about your goals for community moderation *in general*.
  - 👉 You explain why this important to you (e.g. bad behavior on GitHub has lead to key contributors leaving the project).
2. You **charter** a GitHub moderation group to implement that policy.
3. That group writes its own moderation **process** that is *relevant* to GitHub.



*If you now want a moderation group for Discord, with a  
**different process**, you won't need a different  
moderation **policy**.\**

\*This, BTW, is one of my biggest pet-peeve with Contributor Covenant 2.0: it conflates policy and process.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# How this works in practice #2

1. As a group, you write a **policy** about specific goals you want to achieve (the *what*).
2. Someone with domain expertise goes off to write a **process** to implement it.





# Example: Bug triaging

1. The core team agrees triaging bugs needs to improve. It writes a **policy** saying that all bugs must be triaged and assigned to an owner promptly.\*
2. Annie, who does most of triaging, goes off write the specific of the triaging **process**. (*How* issues are labeled, who's responsible for triaging, etc.)

\* You could be more specific here and say something specific (for example: “2 working days”). There's some flexibility here. Experiment and find what works for you. Adapt to circumstances.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



*When you decide to **charter** a dedicated triaging team down the road, it's easy to just **delegate** ownership of the triaging **process** to that team.*

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# How this works in practice #3

1. There are a number of complex **policies** and **processes** to manage IP.
2. This is extremely confusing to new projects who don't know where to start or what's expected of them.
3. You create a **Guidelines** that outlines key requirements and encourages best practices\* (e.g.: OpenJSF IP Guidelines).

\* Worth noting that the policies and processes remain the reference documents.

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)



# Why is this good?

- 🎯 Leadership defines **high-level goals**; those implementing them are **empowered** to do so as *they* see fit.
- 💬 Conversations between leadership and implementers are **naturally structured around this boundary**:
  - *“Is the implementation enabling organizational goals?”*
  - *“Do the goals need to be readjusted?”*



# Why is this good?



**Disagreements are easier to circumscribe:** are you disagreeing about goals and values, or about how to implement them?



**Conversations are focused:** are we discussing *what* we want to achieve or *how*?



Helps **avoid micro-management.**



Creates **autonomy.**



# Why is this good?



It's **flexible**.





It's **maintainable**.



It's **simple**.








## 4. A few additional tips

-  Don't be scared of hard conversations: leveraging governance to engineering yourself out of hard conversations leads to bad outcomes.
-  Manage conflicts 1:1 on a call



# Wrapping up

-  *“Governance is the formalization of implicit norms and culture in order to scale willing collaboration.”*
-  Governance is structured around **authority delegation**.
-  *“Governance is code that has to be interpreted by humans.”*
-  Adopting coding best practices through a **simple framework** to manage authority delegation helps write **flexible and maintainable** governance.
-  Try it and tell me how it works for you!





# Q & A

Tobie Langel, Principal  
[tobie@unlockopen.com](mailto:tobie@unlockopen.com)