# AI, Contribution Policies, and the Rise of Stewardship Communities in Open Source Software

Gregorio Robles[1,*], Daniel M. German[2]

[1]Universidad Rey Juan Carlos, Madrid, Spain
[2]Department of Computer Science, University of Victoria, Victoria, BC, Canada

**Abstract**

Generative and agentic AI are lowering the cost of writing code far faster than they are lowering the cost of deciding whether that code should be merged. This presentation abstract summarizes two companion studies that examine how open source software (OSS) projects are responding to this asymmetry. The first is an empirical analysis of 209 unique AI contribution policies, from which we derive an AI Contribution Governance Framework and a set of role-specific Acceptability Principles for contributor, maintainer, and project. The second paper looks past policy wording at a structural response we call the *stewardship community*: projects that keep using AI internally while closing the external implementation path, retaining broader participation through reports, requirements, and discussion. Read together, the two studies show that AI is not merely a compliance problem to regulate in a CONTRIBUTING file; it is reshaping who gets to become a maintainer, and therefore how OSS communities renew themselves. We propose this presentation to share these findings and open the discussion on evaluation and governance implications with the community.

**Keywords**

Open source software, AI-mediated contribution, contribution policies, code review, governance, stewardship communities, software evolution

## 1. Motivation

Open source contribution has traditionally been *permeable*: anyone can fork a repository, propose a change through a pull request, and, through repeated contribution and review, progress from reporter to contributor to maintainer [3, 4]. Review is the bottleneck that makes this pathway work: it validates a patch, but it also lets maintainers assess a newcomer's judgment and commitment, and lets newcomers learn project conventions. AI coding agents distort this equilibrium. Producing a plausible implementation is now cheap, but understanding someone else's intent, verifying correctness, integrating it, and accepting long-term responsibility for it is not. Pull request volume rises while review capacity stays flat, and the natural effort-based signal that previously separated committed contributors from low-effort submissions is weakened. Our two papers approach the consequences of this asymmetry from complementary angles: one from the bottom up, coding what projects actually write in their policies; the other from the top down, theorizing the organizational form that emerges once maintainers can implement changes themselves faster than they can review an equivalent external contribution.

## 2. Study 1: An Empirical Framework of AI Contribution Policies

We have combined two datasets of OSS AI-related contribution policies (118 projects from Hora and Robbes's study of the 1,000 most-starred GitHub repositories and 106 projects from a community-curated list) into a corpus of 209 unique projects (15 overlap) [1]. Each policy was coded along twelve dimensions (stance, disclosure, human responsibility, scope, enforcement, and others) using an LLM-assisted pipeline validated against manual coding.

Of 209 projects, 42 (20%) prohibit AI-mediated contribution, citing maintainer burden (81%), code quality (79%), and copyright/IP risk (57%), with 19% rejecting AI on ethical grounds. The 34 regulating

*Benevol 2026, Madrid, 26-27 November 2026*
*Corresponding author.
grex@gsyc.urjc.es (G. Robles); dmg@uvic.ca (D. M. German)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

projects (16%) instead condition acceptance on contributor understanding (100%) – the key line separating *regulated* from *prohibited* is whether contributors can explain and defend AI-assisted output. Permissive projects rarely give a rationale at all, focusing instead on procedural requirements like disclosure and testing.

From these findings we derive the *AI Contribution Governance Framework*, which models review as a capacity-constrained signaling problem and organizes recurring concerns (scarcity shift, signal reliability, newcomer pathways, agent accountability) into *Acceptability Principles*, which are explicit conditions for AI-mediated participation, organized around the project, contributor, and maintainer.

## 3. Study 2: Open Source Stewardship Communities

We have focused for this study on a subset of projects that are not merely regulating AI use but closing the external code-contribution path itself, while their own maintainers use AI agents extensively and continue to welcome bug reports, requirements, and discussion [2]. Examples include OpenAI Codex (invitation-only implementation), Ladybird (no public pull requests accepted), Swamp (implementation restricted to the internal team), Symfony Language Tools (issue-first, pull requests disabled), and Ghostty (first contribution requires a maintainer's vouch).

We call this form a *stewardship community*, by analogy with land trusts: a small core retains *title* (i.e, authority over what is implemented and merged) while a broader community contributes through reports and discussion without acquiring title through code contribution. Entry into implementation becomes *gated* rather than self-initiated: selection still happens, but moves earlier, to whoever is admitted through the gate.

This carries governance risks easy to miss from inside a single project. Since existing members control the gate, homophily may narrow future contributor diversity [5]; needs outside the core's priorities are less likely to be implemented; and lost self-initiated contribution concentrates bus-factor risk in a shrinking maintainer pool. A maintainer can do everything human-centric AI asks (e.g., direct agents carefully, retain final authority) while the community still loses its pathway into the core. Keeping humans *in the loop with AI* does not guarantee they stay *in the loop with one another*.

## 4. Contribution of the Presentation

Together, the two papers argue that AI in OSS is not only a code-quality or licensing problem to be solved with a CONTRIBUTING.md clause. It is restructuring the *traditional* pipeline that has turned users into contributors and contributors into maintainers. We hope this presentation will prompt discussion on how tooling, community management practices, and future empirical work can help projects sustain both their software and the people around it.

## References


[1] G. Robles and D. M. German, "You can contribute if you... An Empirical Framework of AI Contribution Policies in OSS," 2026. [Online]. Available: https://arxiv.org/abs/2609.07919

[2] G. Robles and D. M. German, "Open Source Stewardship Communities: 'We need you, but not your pull request'," 2026. [Online]. Available: https://arxiv.org/abs/2609.08456

[3] K. Crowston and J. Howison, "The social structure of free and open source software development," *First Monday*, vol. 10, no. 2, 2005.

[4] B. Trinkenreich, M. Guizani, I. Wiese, A. Sarma, and I. Steinmacher, "Hidden figures: Roles and pathways of successful OSS contributors," *Proceedings of the ACM on Human-Computer Interaction*, vol. 4, no. CSCW2, pp. 1–22, 2020.

[5] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, vol. 27, pp. 415–444, 2001.