Maintaining OSS in the age of AI

Reduce maintainer load, improve quality

Taos Transue (taos.j.transue@gmail.com)
Albuquerque, NM · August 4, 2026

Tragedy of the Commons of Maintainer Attention

The maintainer attention commons is

1. **Shared**
   Contributions draw on the same attention pool.

2. **Finite**
   AI agents are unreliable judges of quality.

3. **Unpriced**
   No contributor pays for what they use.

Increasing contributions exhausts attention.

How can we leverage the available attention to handle the new contribution scale?

Effort was a good quality indicator

Writing software used to come from human effort entirely.

Contributors had to write the patch first

- Test that it worked
- Debug
- Integrate maintainer/community feedback

By integrating feedback, contributors upskilled for future contributions.

AI Agents are driving required effort to zero

WHAT GOT BETTER

- More people can contribute and participate.
- Basic project maintenance is more automatable.

WHAT GOT WORSE

- Huge contribution volume.
- Share of quality contributions.
- Share of contributors who continue contributing.

Reviewing used to have two benefits: a better project and, a new trained colleagues. As contribution noise increases, payoff for review effort falls.

Towards a scalable review process

We need machinery to filter and sort it.

1 Pace incremental improvements

Generate the small improvements yourself at a comfortable pace.

2 Filter AI noise

Ensure requests for maintainer attention got human contributor attention.

3 Improve what arrives

Aggregate the quality parts of your contributors' PRs.

Slow loops

A *small* agent pull request a day.

EXAMPLE SLOW LOOPS

Check documentation against the current implementation.

Small code refactors:

- **Code API consistency**

“Drive-by” pull requests:

- **E.g., Typo fixes, etc.**

You're not blocking contributors with a genuine interest in your project.

Context engineering with Dex Horthy. https://newsletter.pragmaticengineer.com/p/context-engineering-with-dex-horthy

Filtering AI agent noise

Find PRs with no human effort with an ensemble of *simple filters*.

Automatically detecting AI code is a tricky arms race.

- We want contributors that try, regardless of their AI usage.

THREE FILTERS, WEAKEST TO STRONGEST

1. **PR template checkbox**
   Agents might fail this now.

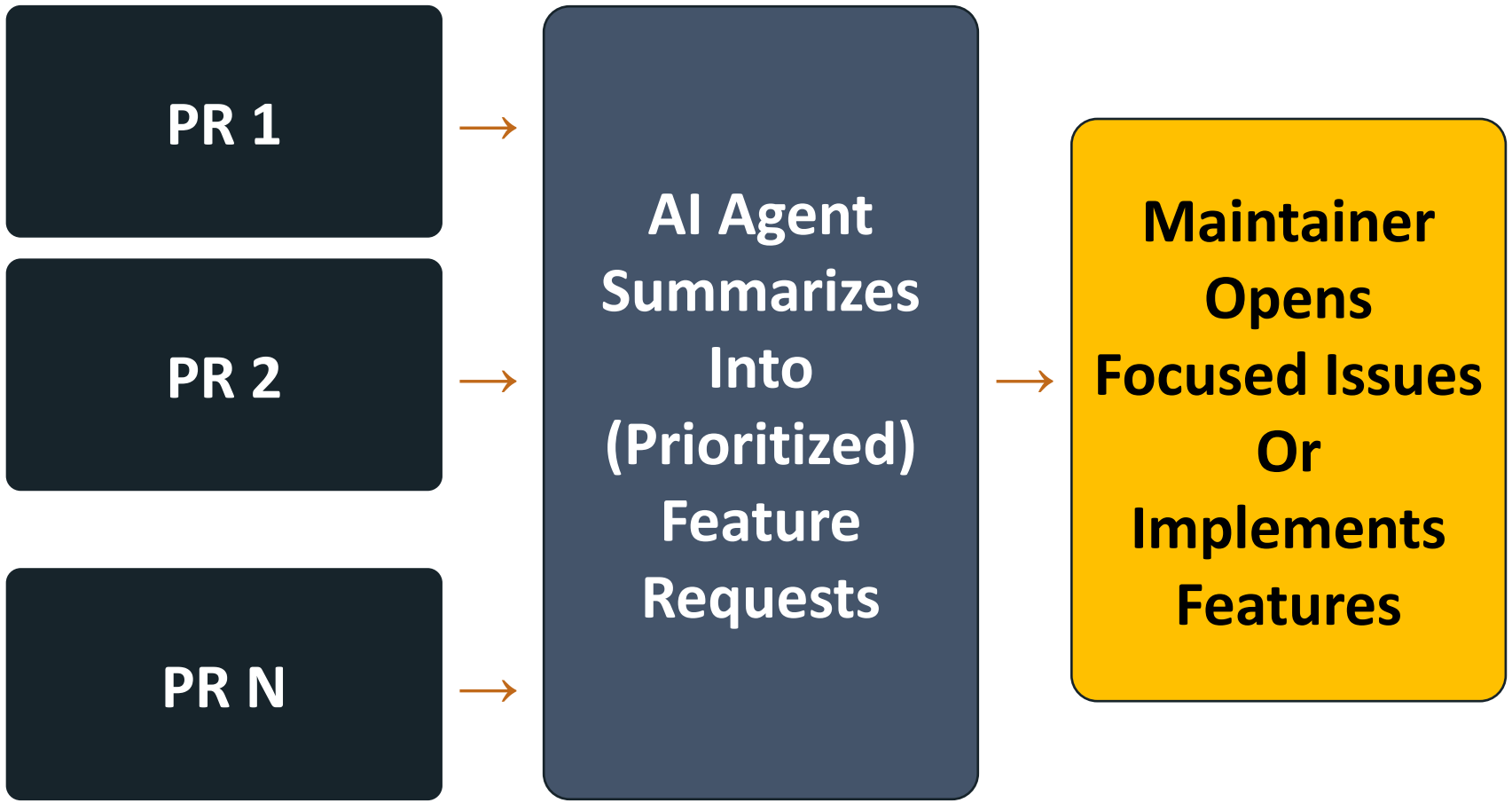
2. **The disclosure file**
   Humans open PRs only. If an agent does, it must disclose so in the diff.

3. **PRs reference an open issue**
   Only maintainers create issues.

Mitchell Hashimoto's new way of writing code. https://newsletter.pragmaticengineer.com/p/mitchell-hashimoto

Improving what PRs arrive; no more merging PRs?

PRs may become the new method of feature requests.



Supporting your users using AI agents

Introduction to writing AI agent skills

AI hallucination-driven development

- AI has read lots and lots of code.
- If agents are hallucinating a particular API in your project,
  - consider implementing it.

A skill is saved context you and the agent can call

Temporary training of an agent, delivered through context.

`SKILL.md`

Name, description	What and when.
Body	Main instructions.
References	Loaded on demand.

SKILL CONTENT

Capability

Teach the agent something it can't do reliably yet.

Preference

Project preferences: review checklists, style guides. AGENTS.md

Philipp Schmid. Don't Ship Skills Without Evals
https://www.youtube.com/watch?v=0vphxNt4wyk

The two skills to support users

1 Shared vocabulary

Define your project's domain terms for the agent.

- Improves user/agent alignment

2 Interview, then plan

Interview the user on goals

↓

Generate implementation plan

Matt Pocock. Building Great Agent Skills: The Missing Manual
https://www.youtube.com/watch?v=UNzCG3lw6O0

Writing effective skills

1. **A good description is critical**
   Determines if the skill is used.
   - Need WHAT and WHEN
2. **Give clear direction**
   No passive “we recommend”.
3. **200--500 lines long**
   Reference to longer text in the skill.

34% → 50%

Task success rate improvement with skills. (SkillsBench)

Skills drift like documentation.

Check alignment with slow loops.

Matt Pocock. Building Great Agent Skills: The Missing Manual.
https://www.youtube.com/watch?v=UNzCG3lw6O0
Philipp Schmid. Don't Ship Skills Without Evals.
https://www.youtube.com/watch?v=0vphxNt4wyk