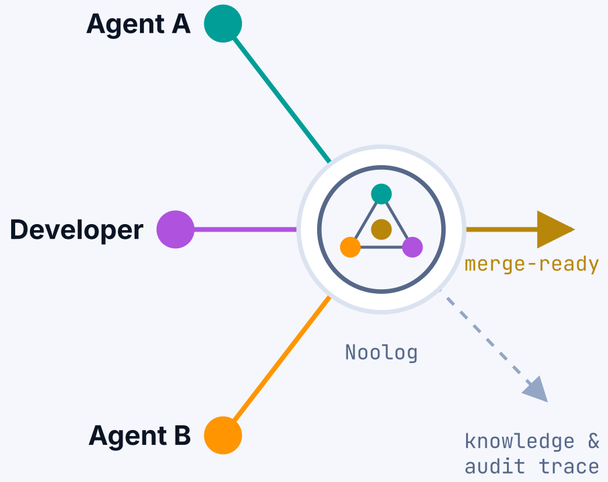
Noolog

FOR LEAN TEAMS SHIPPING WITH CODING AGENTS

Next-gen code review pipeline.
Ship in parallel, with confidence.

Noolog is a collective intelligence platform that makes code merge-ready.

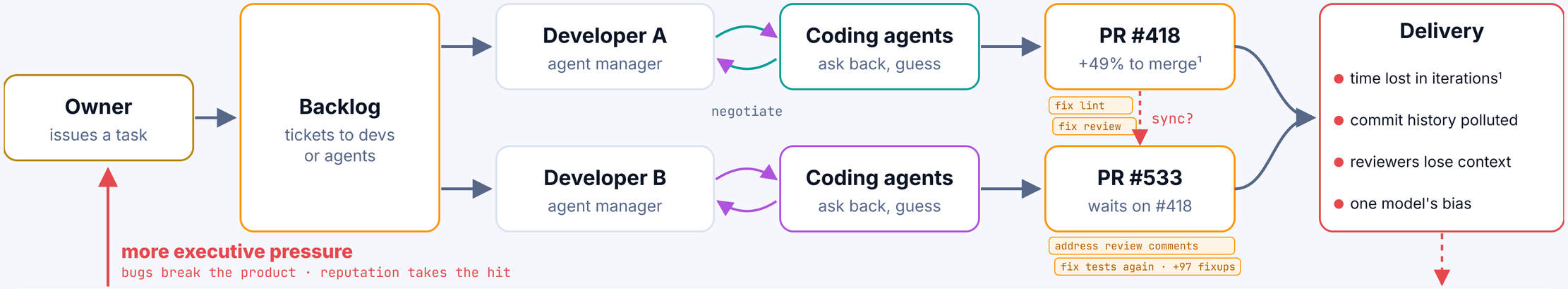
> *"github creates absolutely useless garbage merges ... Linux kernel merges need to be done **properly**."*
>
> © Linus Torvalds, creator of Linux and Git · LKML, 2021



THE PROBLEM

Agents write fast. Project pipelines can't keep up.

Filing tickets for every change **either loses tempo or loses purpose.**



COMPREHENSION LOST · WHAT IT COSTS

17.6 h
first review, agent PRs[2] backlog waits

33%
of AI PRs ever merge[3] the rest is rework

25%
of incoming PRs are AI slop[4] Linux Foundation, 2026

39.9%
of AI-commit issues pass[5] human review to production

44%
of AI tasks add a vulnerability[6]

Comprehension debt[7]
code nobody on the team can explain anymore

WHAT SLIPS THROUGH COSTS

AWS · 13-hour outage[8]
AI agent recreated production

rsync · backups broken[9]
AI-assisted release, emergency fix

AND WHO CHECKS?

One model, one opinion
no second perspective from other models before code ships

LINUX KERNEL INSPIRED FRAMEWORK

Parallelized production for excellence.



4 in 10

misses caught in milliseconds

First findings in seconds, on 5,173 real repos.[5]



~9×

more blockers caught

Less rework than one AI reviewer.[11]



Spec first

shorter lead time

Acceptance criteria agreed before code.



Multiplayer

agile stand-ups, automated

Confidence builds with every stakeholder involved.



+82%

same models, as a council

Models from different labs check each other: ~40% less sycophancy.[12]



85%

of code smells caught before commit

Fewer correction loops. A lower bound: CI already stops broken code.[5]

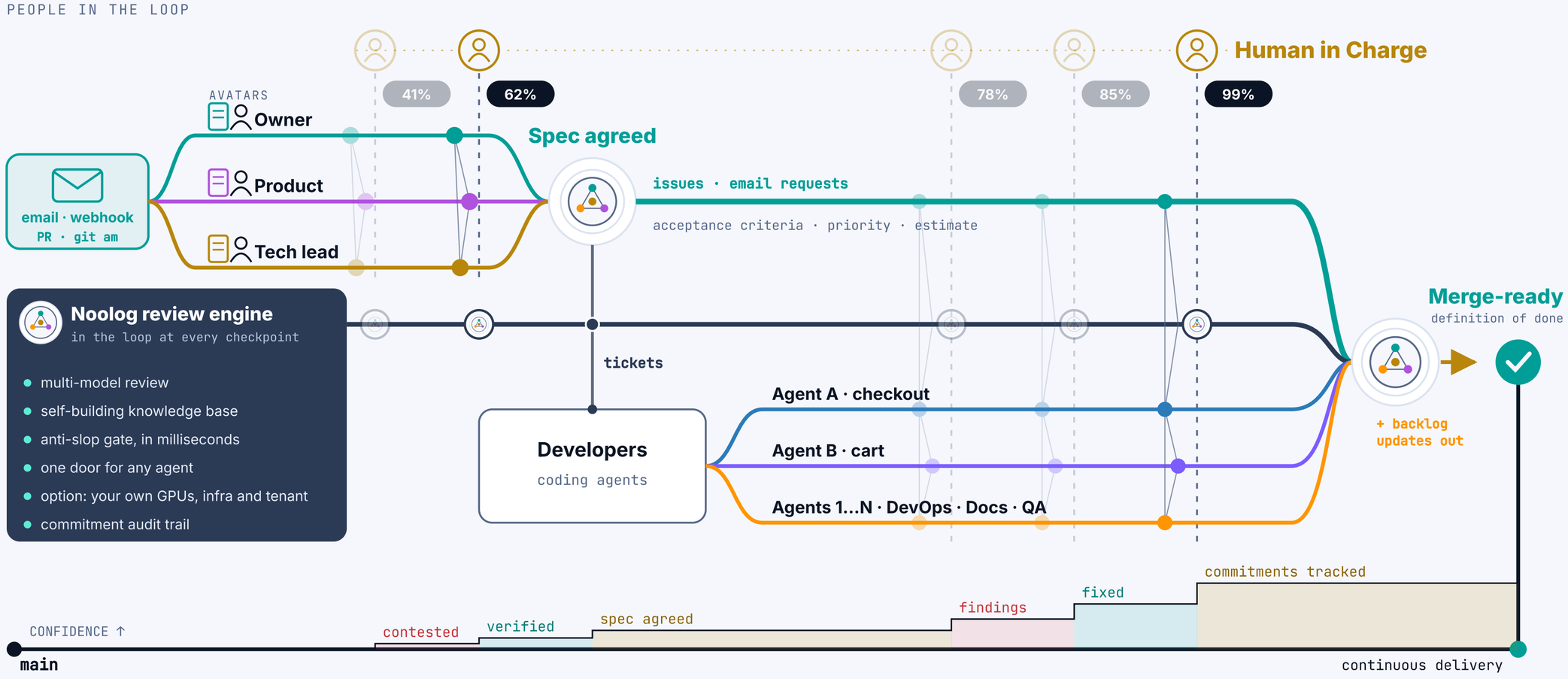


5 sloppoke-bench · 10 kernel guide · 11 kernel review benchmark · 12 arXiv 2601.16863 · example confidence

THE FLOW

High-speed rails, built for a new era of development.

Linux kernel methodology at hand, Just-In-Time. Replace stand-ups with checkpoints. **Finish sprints retrospective-ready.**



Example confidence values

NOOLOG REVIEW ENGINE

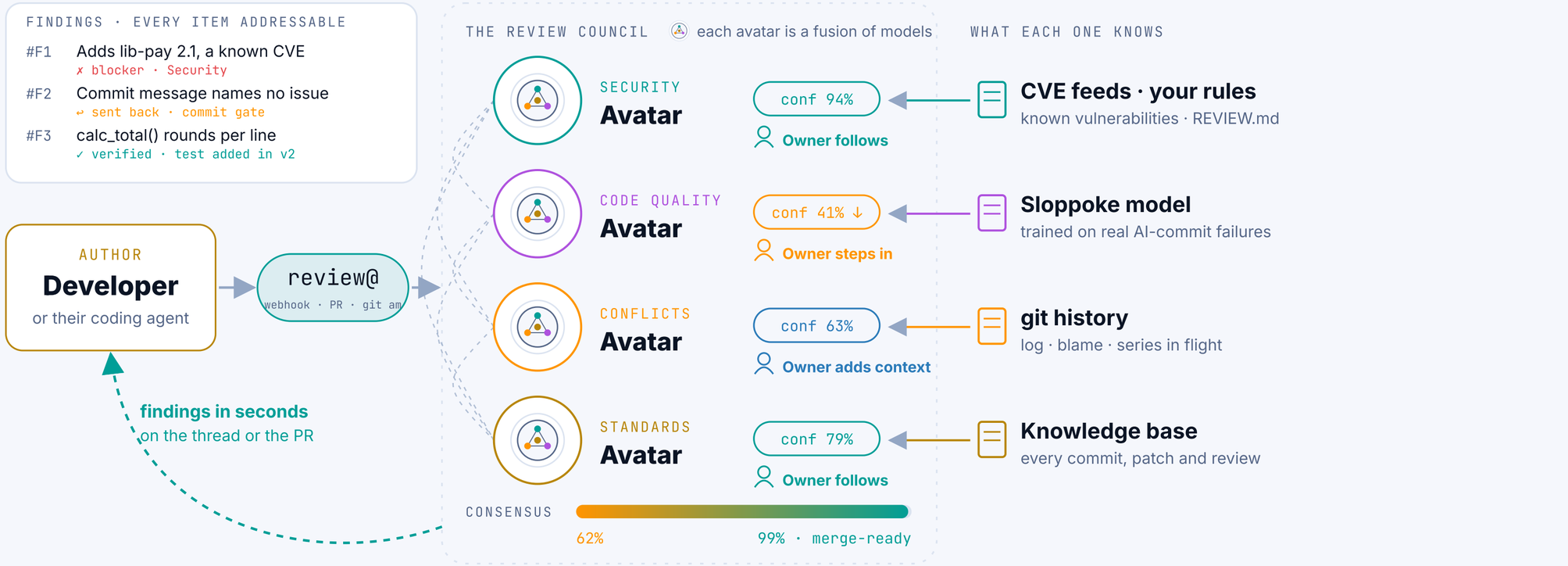
One patch. Many perspectives.

1
A patch arrives.
By email, webhook or your PR.

2
Applied with git am
onto your base.

3
Avatars review
with multiple models and data sources.

4
People step in
only when confidence falls.



Confidence values from an example review

CATEGORIES, NOT VENDORS

Beside your tracker. Ahead on review.

	AI PR reviewers	Trackers with built-in agents	Coding agents on their own	Mailing-list development	Noolog
No single-model bias	—	—	—	—	●
First findings in seconds, refined at every checkpoint	◐	◐	—	◐	●
Conflicts between parallel agents caught before merge	—	—	—	◐	●
Spec agreed by stakeholders before code	—	◐	◐	—	●
Clean, indexable commit history	—	—	—	●	●
Knowledge that stays with the team	◐	●	—	◐	●
Executives and stakeholders kept in sync	—	◐	—	—	●
Backlog, boards and reporting	—	●	—	—	◐
No pull request required	—	—	—	●	●

● yes ◐ partial — no

HOW TO GET STARTED

Pilot and deployment.

01 · CODE REVIEW — ~30 minutes

Every commit reviewed

1. Shadow mode: runs beside your current review, blocks nothing
2. Point git at it, or your PR webhook

```
git config sendemail.to review@your-team.noolog.io
```

3. After 2 weeks: a report of what it caught that you missed

Your pull requests keep working while you decide.

02 · FRAMEWORK — week 1

Tasks in, merge-ready out

1. Send a task by email, or point your PM tool's webhook at Noolog
2. Name your stakeholders; their avatars learn your docs and history
3. The first spec checkpoint lands in your inbox; agents pick up the tickets

Spec agreed before code. The backlog updates itself.

03 · DEPLOYMENT — your choice

Run it where you are

1. SaaS

```
managed by us, EU cloud
```

2. Managed application

```
Azure · Alibaba Cloud · AWS
```

3. Your own servers

```
on-prem or air-gapped
```

Same engine, same audit trail, everywhere.

Pilot measured by cycle time · lead time · rework · escaped defects

tim@peeramid.xyz

SOURCES

Every number, traceable.

1. Harvard study of coding-agent adoption
https://complexdiscovery.com/harvard-researchers-find-ai-coding-agents-shifted-the-bottleneck-to-code-review/
2. LinearB: agent PRs wait 17.6 h for a first review, vs 3.4 h
https://hackernoon.com/ai-made-writing-code-cheap-getting-it-reviewed-now-takes-5x-longer-says-linearbs-ceo
3. LinearB, 8M+ pull requests
https://linearb.io/library/ai-in-software-development
4. Linux Foundation: 25% of incoming PRs are AI slop, Oct 2026
https://www.linuxfoundation.org/press/new-lf-research-survey-shows-generative-ai-cuts-open-source-coding-time-in-half-but-25-of-incoming-pull-requests-are-considered-ai-slop
5. sloppoke-bench, 5,173 OSS repos
https://github.com/peeramid-labs/sloppoke-bench
6. Veracode GenAI Code Security Report 2026
https://www.veracode.com/blog/2026-genai-code-security-report-ai-risk/
7. Comprehension Debt in GenAI-Assisted Software Engineering Projects, EASE 2026
https://arxiv.org/html/2604.13277
8. The Guardian via FT: AWS outages and AI tools, Feb 2026
https://www.theguardian.com/technology/2026/feb/20/amazon-cloud-outages-ai-tools-amazon-web-services-aws
9. Pivot to AI: rsync release breaks backups, Jun 2026
https://pivot-to-ai.com/2026/06/03/rsync-goes-ai-slop-breaks-your-backups/
10. Linux kernel maintainer handbook: creating pull requests
https://www.kernel.org/doc/html/v4.18/maintainer/pull-requests.html
11. Kernel patch-series review benchmark
https://gist.github.com/peersky/1be644c434f58ca10483cf860b316438
12. Noolog benchmarks, arXiv 2601.16863
https://arxiv.org/abs/2601.16863
Title quote: Linus Torvalds on LKML, Sep 2021
lore.kernel.org/lkml/CAHk-=wjbtip559HcMG9VQLGPmkurh5Kc50y5BceL8Q8=aL0H3Q@mail.gmail.com

Our own figures are reproducible: the bench repository and the paper carry the scripts and data