aviator

How to kill the code review

The gate is still there. It just isn't stopping anything.

Ankit Jain

aviator

I'm going to argue we should kill the code review

aviator

I'm going to argue we should kill the code review

I run a company that sells the replacement

aviator

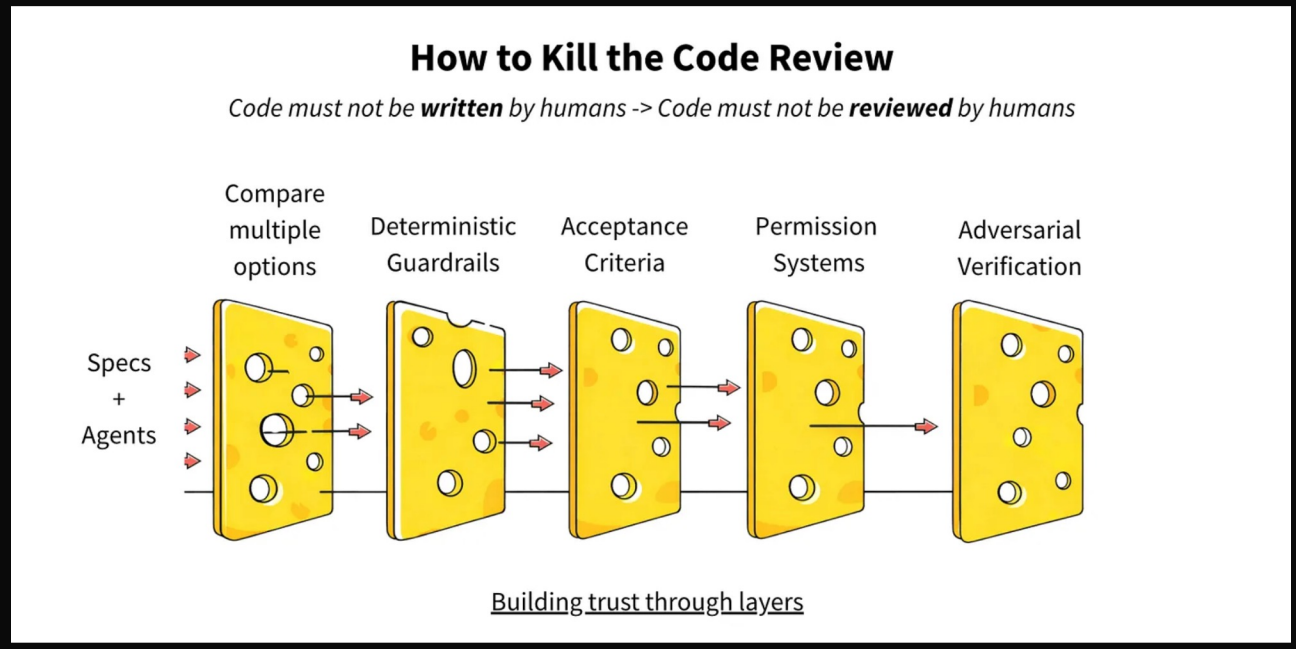
I'm going to argue we should kill the code review

I run a company that sells the replacement

So be suspicious of me

aviator

Code reviews are dead



latent.space/p/reviews-dead

Latent Space · March 2026

aviator

Five layers of trust

latent.space/p/reviews-dead

aviator

Five layers of trust

1 **Compare multiple options**

several agents try it — rank by tests, diff size, deps

latent.space/p/reviews-dead

aviator

Five layers of trust

1 **Compare multiple options**

several agents try it — rank by tests, diff size, deps

2 **Deterministic guardrails**

linters, types, contracts. Facts, not opinions

latent.space/p/reviews-dead

aviator

Five layers of trust

1. **Compare multiple options**
   several agents try it — rank by tests, diff size, deps
2. **Deterministic guardrails**
   linters, types, contracts. Facts, not opinions
3. **Humans define acceptance criteria**
   written upstream, before the code exists

latent.space/p/reviews-dead

aviator

Five layers of trust

1. **Compare multiple options**
   several agents try it — rank by tests, diff size, deps
2. **Deterministic guardrails**
   linters, types, contracts. Facts, not opinions
3. **Humans define acceptance criteria**
   written upstream, before the code exists
4. **Permission systems**
   scope what an agent may touch unattended

latent.space/p/reviews-dead

aviator

Five layers of trust

1. **Compare multiple options**
   several agents try it — rank by tests, diff size, deps
2. **Deterministic guardrails**
   linters, types, contracts. Facts, not opinions
3. **Humans define acceptance criteria**
   written upstream, before the code exists
4. **Permission systems**
   scope what an agent may touch unattended
5. **Adversarial verification**
   a separate agent tries to break it

latent.space/p/reviews-dead

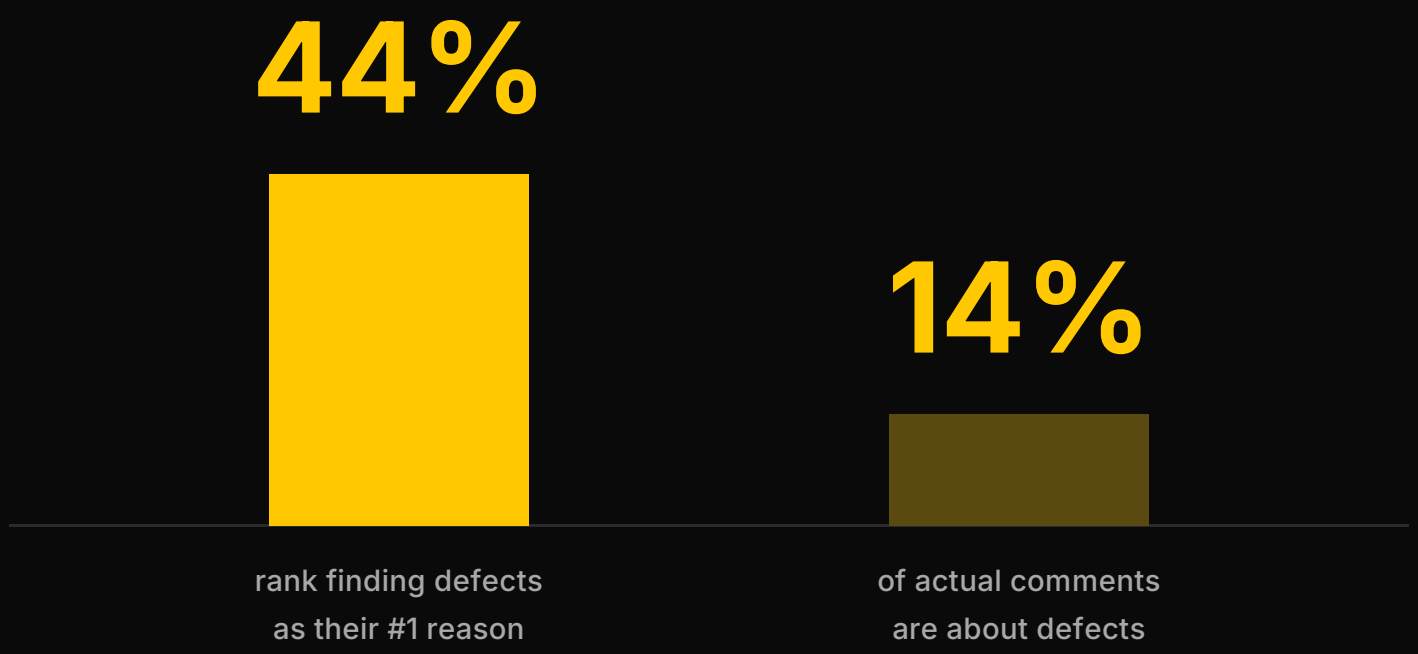
aviator

01

Why do we review code

aviator

44% say it's to find defects



Bacchelli & Bird · ICSE 2013 · 570 comments classified

aviator

9,000,000

reviewed changes at Google

same finding

Sadowski et al. · ICSE-SEIP 2018

aviator

Code review is not about the code

aviator

Code review is not about finding bugs

aviator

Code review is to build mental models

Code review is for knowledge sharing

aviator

We made writing cheap, and understanding stayed exactly as expensive as it has always been

— Addy Osmani

aviator

02

The problem

How long until we stop reading the code?

aviator

We already have

Faros AI Engineering Report 2026 · 12 Apr 2026 · 22,000 devs · 4,000+ teams

aviator

We already have

+242.7%
incidents per PR

+54%
bugs per developer

+31.3%
PRs merged with no review at all

+13.8%
work restarts

Faros AI Engineering Report 2026 · 12 Apr 2026 · 22,000 devs · 4,000+ teams

aviator

AI adoption raises delivery throughput

And delivery instability

At the same time

DORA · State of AI-assisted Software Development 2025

aviator

Those devs who put the same effort and energy into code review as before feel overloaded by AI slop PRs sent their way

— Gergely Orosz

aviator

No one voted to stop reviewing code

We just stopped

aviator

No one voted to stop reviewing code

We just stopped

aviator

Can AI review the code?

aviator

“AI sucks at reviewing code”

You get 15 findings, 12 are noise,

now you research each one to find the 3 that matter

aviator

But we were never that good at it either

aviator

Because a diff is a bad place for review

One reviewer, reading once, from memory

aviator

Because a diff is a bad place for review

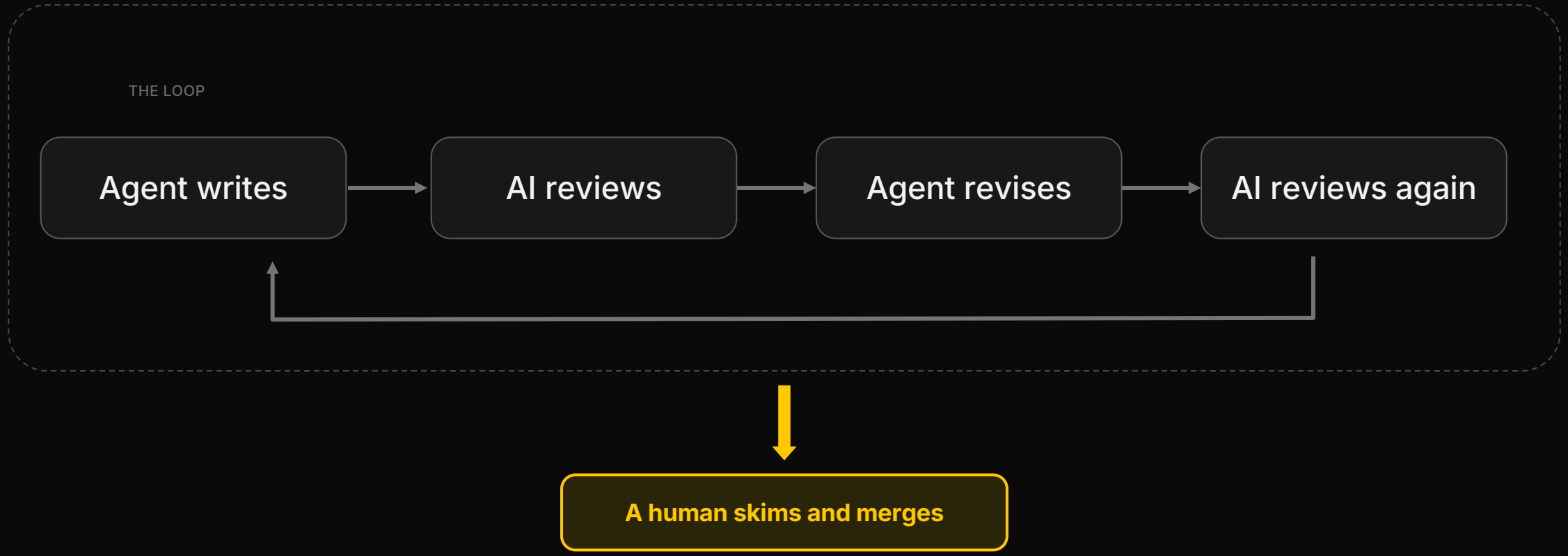
One reviewer, reading once, from memory

aviator

Instead, we built review theater

aviator

Instead, we built review theater



aviator

Can AI genuinely review code well?

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

Can AI genuinely review code well?

THE CASE FOR

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

Can AI genuinely review code well?

THE CASE FOR

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

THE CASE AGAINST

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

Can AI genuinely review code well?

THE CASE FOR

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

THE CASE AGAINST

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

Can AI genuinely review code well?

THE CASE FOR

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

THE CASE AGAINST

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

Can AI genuinely review code well?

THE CASE FOR

It never gets tired, or rushed

It reads every line, every time

Same standard on every PR

It is awake at 3am

THE CASE AGAINST

It never saw what was decided

It was trained on the code it is reviewing

It will never tell you not to build it

It cannot be accountable

aviator

But a diff can't tell you what was decided

aviator

Build tools for the things AI does well

Protect the things it cannot

We have been doing the exact opposite

aviator

Not everyone agrees

In the era of AI coding, human review is being scrutinized as a bottleneck to writing software. While that may be true, we have no current interest in removing that bottleneck.

— Wealthfront Engineering

3 Aug 2026 · eng.wealthfront.com/2026/08/03/experiments-with-ai-code-review

aviator

They're right

The replacement doesn't exist yet

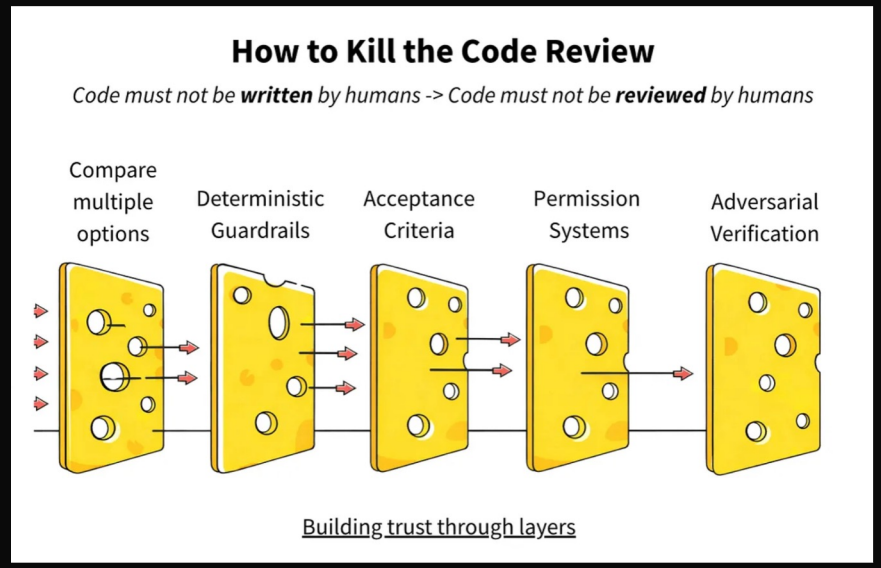
03

Review of the essay

That was six months ago. In AI time that's a decade.

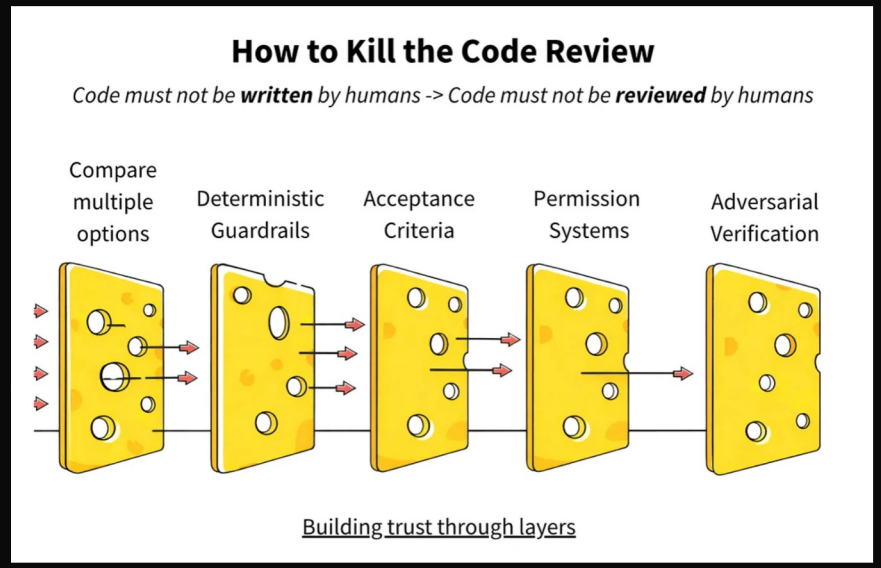
aviator

Five layers of trust



aviator

Five layers of trust



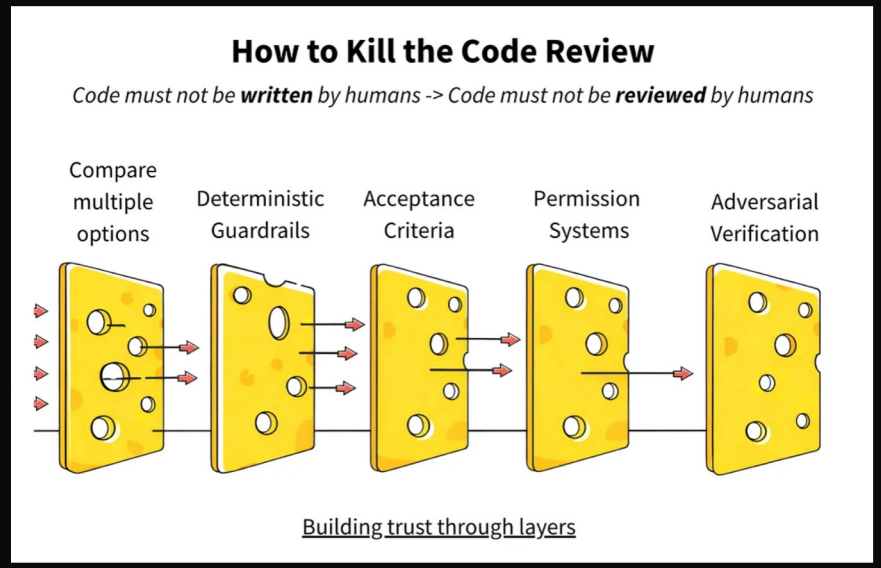
Four layers answered “Is it right?”

One answered “Is it what we meant?”

None of them asked

“Should we have built it?”

Five layers of trust



Four layers answered “Is it right?”

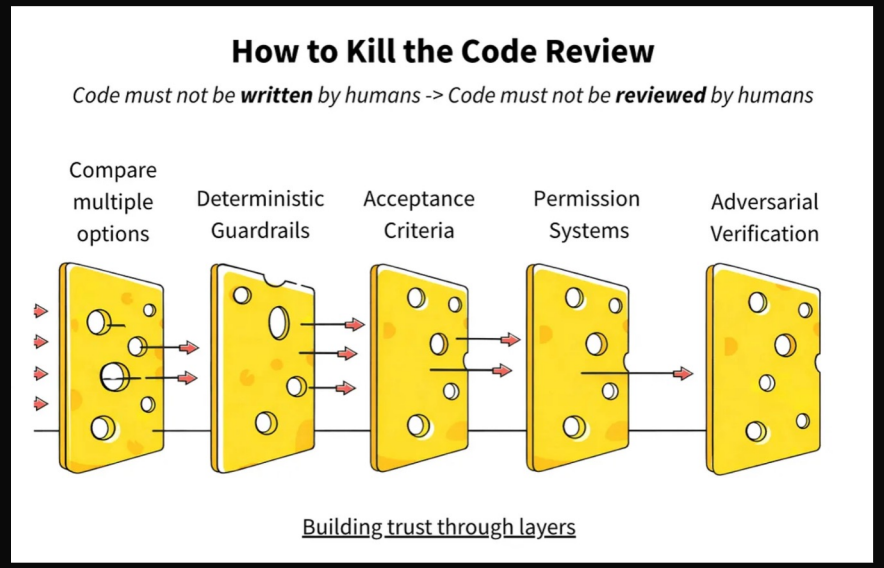
One answered “Is it what we meant?”

None of them asked

“Should we have built it?”

aviator

Five layers of trust



Four layers answered “Is it right?”

One answered “Is it what we meant?”

None of them asked

“Are we building the right thing?”

aviator

The structure was right

I missed the human judgment entirely

aviator

04

The framework

Same concepts. Better order. Practical implementation.

aviator

Five concepts, four corrections

THE ESSAY · MARCH 2026	WHAT IT BECOMES IN PRACTICE
Adversarial verification	1 · Argue (moved to the front)
Humans define acceptance criteria	2 · Capture (widened to include intent)
Deterministic guardrails	3 · Codify (the AI slop register)
Compare multiple options	folds into Argue
Permission systems	5 · Ownership

aviator

Five concepts, four corrections

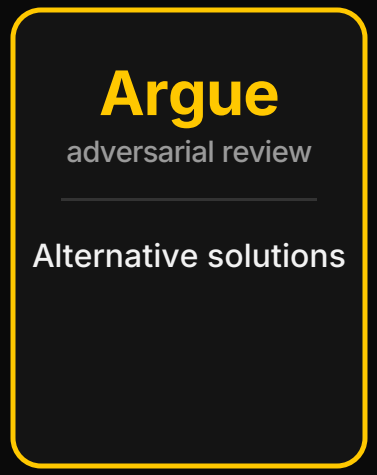
THE ESSAY · MARCH 2026	WHAT IT BECOMES IN PRACTICE
Adversarial verification	1 · Argue (moved to the front)
Humans define acceptance criteria	2 · Capture (widened to include intent)
Deterministic guardrails	3 · Codify (the AI slop register)
Compare multiple options	folds into Argue
Permission systems	5 · Ownership
???	4 · Debate — the one that wasn't there

aviator

The new shape of review

aviator

The new shape of review



aviator

The new shape of review

Argue
adversarial review

Alternative solutions

Capture
intent + criteria

Knowledge transfer

aviator

The new shape of review

Argue
adversarial review

Alternative solutions

Capture
intent + criteria

Knowledge transfer

Codify
the slop register

Finding defects

Consistency

aviator

The new shape of review

Argue	Capture	Codify	Debate
adversarial review	intent + criteria	the slop register	the decisions
Alternative solutions	Knowledge transfer	Finding defects Consistency	— new —

aviator

The new shape of review

Argue	Capture	Codify	Debate	Own
adversarial review	intent + criteria	the slop register	the decisions	the signature
Alternative solutions	Knowledge transfer	Finding defects Consistency	— new —	Gatekeeping

aviator

1 Argue | 2 Capture | 3 Codify | 4 Debate | 5 Own

Let the agents argue before the PR exists

aviator

Why is there a UI for two machines talking to each other?

aviator

Why is there a UI for two machines talking to each other?

Let them argue

before the PR is created

aviator

Instead of reading 4000 lines

Read what they disagreed about

aviator

Instead of reading 4000 lines

Read what they disagreed about

aviator

Layer 1 – ARGUE

Capture the decisions not the verdict

aviator

What they couldn't settle

is what you debate

aviator

You can do this today

Run two reviewers, different models

Write a Skill that captures the disagreements

Review that as part of the PR

aviator

Move AI reviews upstream

PR-Agent

Two models on the PR. GPT, Claude, Gemini, DeepSeek or local via Ollama

MIT · 12.8k stars

github.com/The-PR-Agent/pr-agent

Aider — architect mode

One model proposes, a different one edits. Before the PR exists

Apache-2.0 · 48.6k stars

aider.chat/docs/usage/modes.html

AutoGen

Multi-agent conversation and group orchestration

MIT · 60.7k stars

github.com/microsoft/autogen

CrewAI

Role-based agents, if you want to define the arguing sides yourself

MIT

github.com/crewAIInc/crewAI

aviator

1 Argue | 2 Capture | 3 Codify | 4 Debate | 5 Own

Capture the what and the why

aviator

The popular answer is to write the spec up front

aviator

Why specs fail?

Requires time

There's no feedback loop

The implementation drifts

That's the waterfall-model we discarded 20 years ago

aviator

Why specs fail?

Requires time

There's no feedback loop

The implementation drifts

That's the waterfall-model we discarded 20 years ago

aviator

What can replace spec?

Intent - why are we making this change?

Acceptance criteria — how should it behave?

These are the decision artifacts

code is build output

aviator

What can replace spec?

Intent - why are we making this change?

Acceptance criteria - how should it behave?

These are the decision artifacts

code is build output

aviator

What can replace spec?

Intent - why are we making this change?

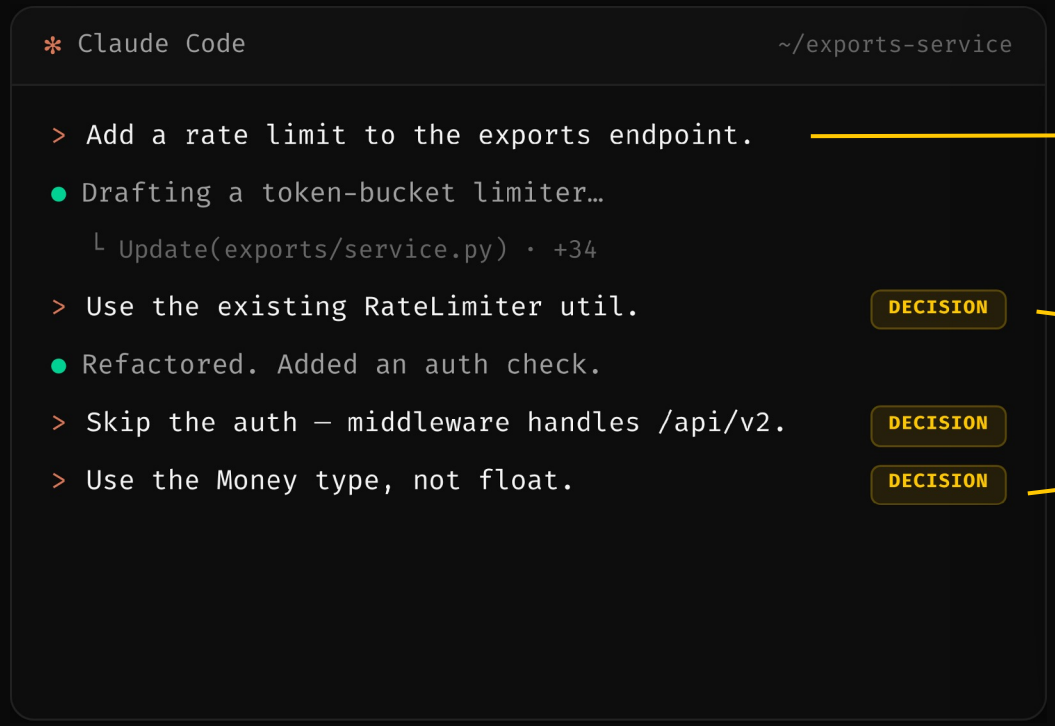
Acceptance criteria - how should it behave?

Works for a 1-line bug fix

and for a 1000-lines feature

aviator

Where do these live?



intent

acceptance criteria

And we throw away the decisions when the session ends

aviator

Do this today

Capture intent and criteria during the session

Publish them on the pull request

so the reviewer reads what you decided

aviator

1 Argue | 2 Capture | 3 Codify | 4 Debate | 5 Own

Turn repeated comments into deterministic rules

aviator

Two kinds of review comment

"Have you considered doing this differently?"

"Use the Money type, not float."

aviator

Two kinds of review comment

"Have you considered doing this differently?"

"Use the Money type, not float."

Only one of them needs human judgment

aviator

CODIFY - The AI slop register

The running list of patterns
your team keeps correcting

aviator

Codified as invariants

aviator

Codified as invariants

- Money type for anything with a currency
- No direct writes to the users table
- Structured logger only — no print
- Every error path emits a counter

aviator

Standing rules
checked on every change
Not remembered. Enforced.

Every review comment your team repeats is a guardrail you haven't written

aviator

1 Argue | 2 Capture | 3 Codify | **4 Debate** | 5 Own

Humans, on the decisions

aviator

No verification layer answers this

aviator

Not one of my five

aviator

And here's the part that worries me

aviator

Code review is the only scheduled moment
engineers talk to each other
about the system

Automate code review, and you delete shared understanding

aviator

The death of a program happens when the programmer team possessing its theory is dissolved

— Peter Naur, 1985

Programming as Theory Building · verified verbatim

aviator

Knowledge sharing was never a side effect of review

It was THE PRODUCT

aviator

Knowledge sharing was never a side effect of review

IT WAS THE PRODUCT

aviator

1 Argue | 2 Capture | 3 Codify | 4 Debate | 5 Own

What happens to accountability?

aviator

A model has no reputation

A model has no liability

A model cannot be asked why

aviator

Responsibility moved

aviator

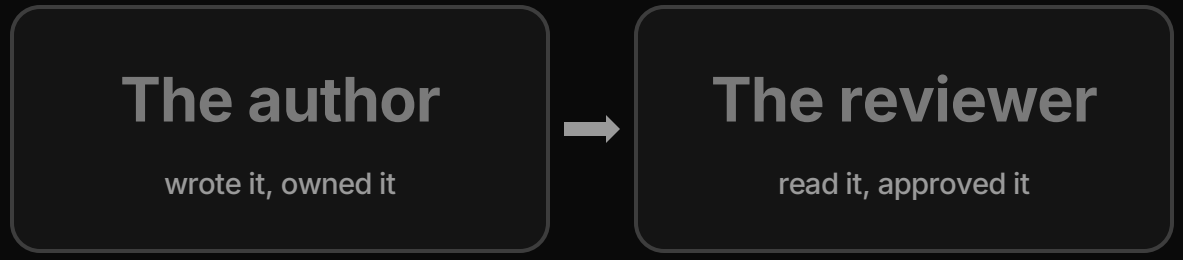
Responsibility moved

The author

wrote it, owned it

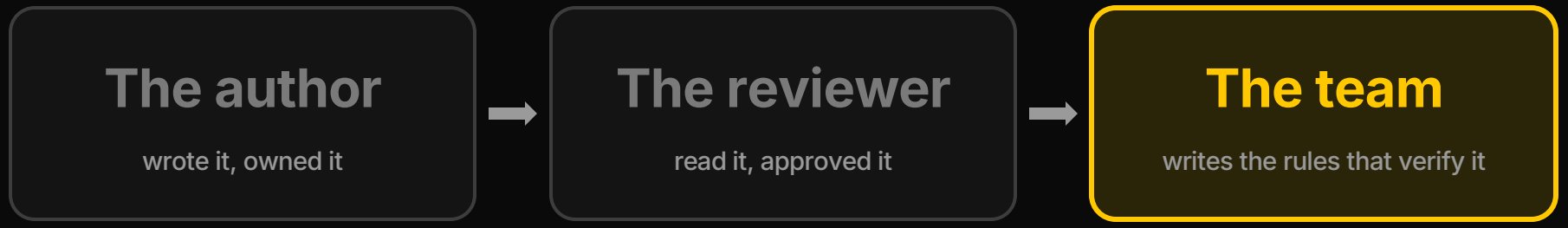
aviator

Responsibility moved



aviator

Responsibility moved



aviator

What ownership means now

Our job is not line-by-line review

It's owning the rules that govern the code

and maintain the mental model behind them

aviator

What ownership means now

Our job is not line-by-line review

It's owning the rules that govern the code

and maintain the mental model behind them

aviator

What ownership means now

Our job is not line-by-line review

It's owning the rules that govern the code

and maintain the mental model behind them

aviator

05

Here’s your homework

aviator

Kill the code reviews!

aviator

~~Kill the code reviews!~~

Please don’t

aviator

Kill the line-by-line review theater

aviator

AI is good at finding bugs

Humans are good at judgment

aviator

Five layers, five things

aviator

Five layers, five things

1 **Run a second reviewer**

different model — PR-Agent is MIT and free

aviator

Five layers, five things

1. **Run a second reviewer**
   different model — PR-Agent is MIT and free
2. **Write down what they settled**
   intent and acceptance criteria, on the PR

aviator

Five layers, five things

1. **Run a second reviewer**
   different model — PR-Agent is MIT and free
2. **Write down what they settled**
   intent and acceptance criteria, on the PR
3. **Harvest your last 1000 review comments**
   the top 20 become invariants

aviator

Five layers, five things

1. **Run a second reviewer**
   different model — PR-Agent is MIT and free
2. **Write down what they settled**
   intent and acceptance criteria, on the PR
3. **Harvest your last 1000 review comments**
   the top 20 become invariants
4. **Change what the conversation is about**
   decisions, not the diff

aviator

Five layers, five things

1. **Run a second reviewer**
   different model — PR-Agent is MIT and free
2. **Write down what they settled**
   intent and acceptance criteria, on the PR
3. **Harvest your last 1000 review comments**
   the top 20 become invariants
4. **Change what the conversation is about**
   decisions, not the diff
5. **Define ownership of rules**
   and what ownership now means

aviator

Work on the tools and culture together

Build tools to push decisions to the same PR

Have a conversation with the team to change behavior

Define strong guardrails to build the trust

aviator

Work on the tools and culture together

Build tools to push decisions to the same PR

Have a conversation with the team to change behavior

Define strong guardrails to build the trust

aviator

Work on the tools and culture together

Build tools to push decisions to the same PR

Have a conversation with the team to change behavior

Define guardrails to build the trust

Code review is not about the code

aviator.co/verify

You can outsource your thinking,

but you can't outsource your understanding

— Andrej Karpathy