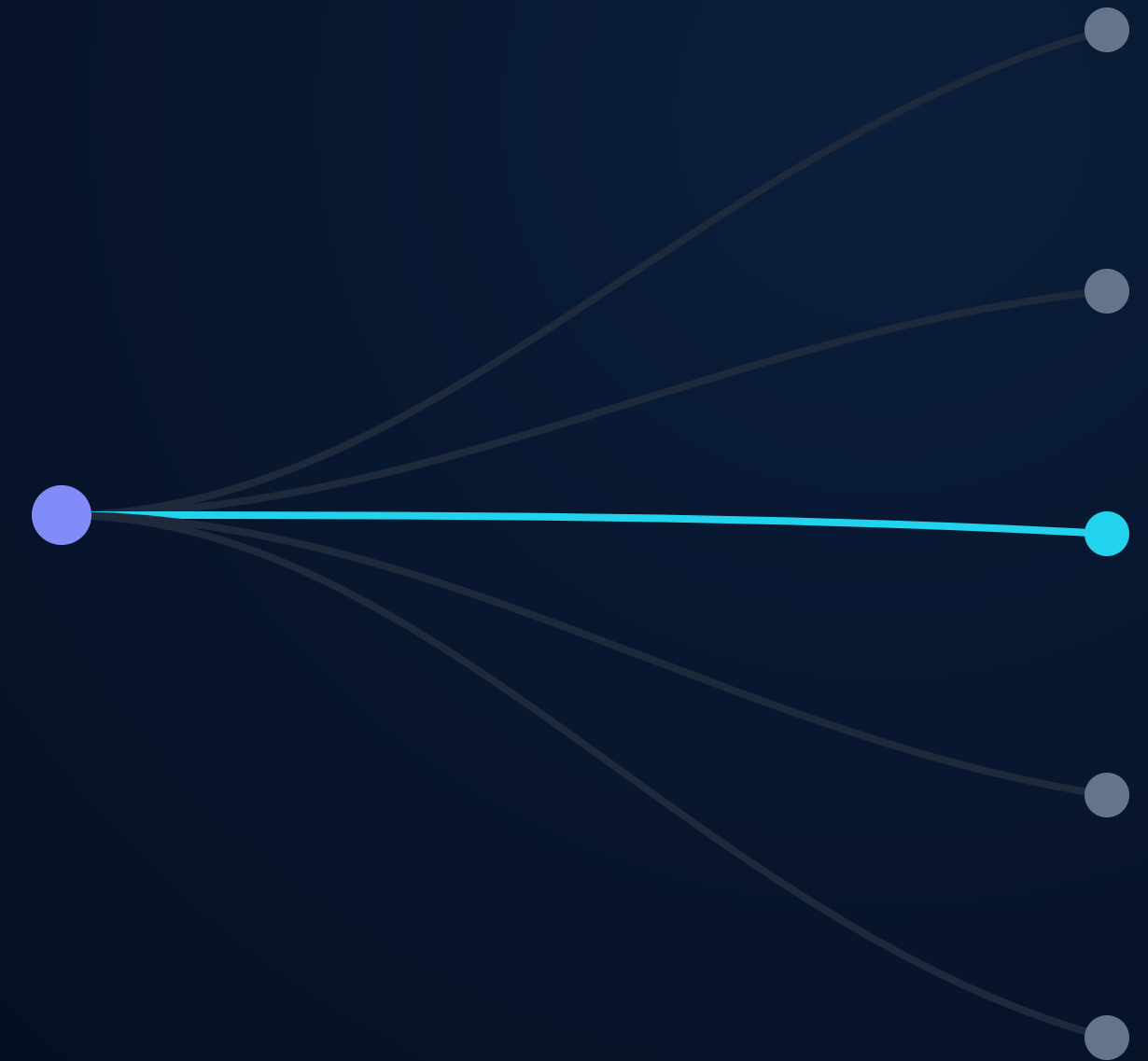


One Team, Many AIs

Keeping engineering teams aligned as AI workflows pull them apart

Pablo Ibanez



One team. Five developers. Five AI workflows.

DEV 1

Cursor with
custom rules

DEV 2

Claude Code with
an instruction file

DEV 3

Copilot in the
editor

DEV 4

Custom agents
and integrations

DEV 5

ChatGPT, copy
and paste

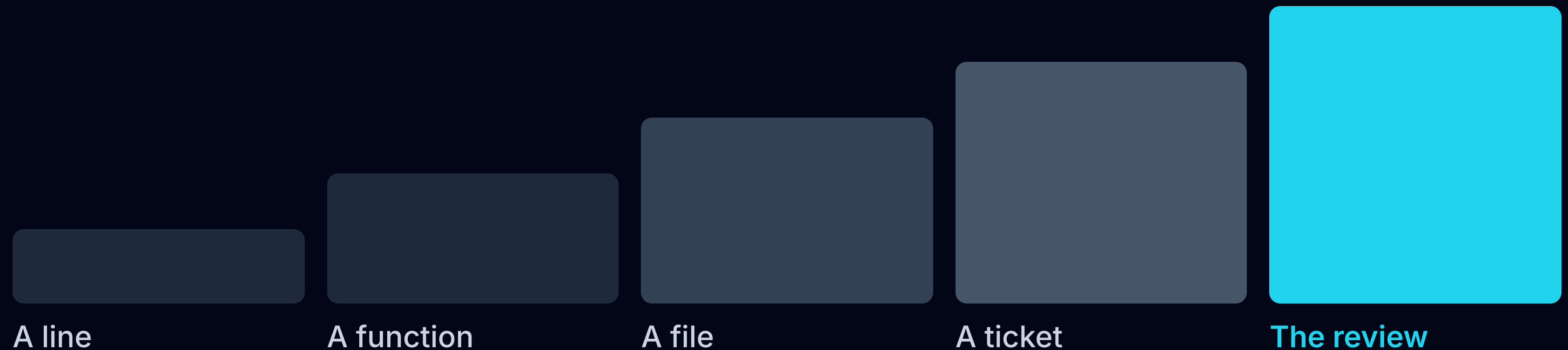
+ REVIEW ...and one AI reviewing all of it.

Same repository · same sprint · same standards

Are they still one team?

CONTROL IS SHIFTING

Nobody decided to hand over control.



It happened one accepted suggestion at a time.

So who's really in control?

01

A new kind of drift in our teams

02

Why it matters more, now that AI reviews our code

03

What we can do, without limiting experimentation

THE NEW BOTTLENECK

We write faster than we can review.

So teams review less carefully, or let AI review.

PR #412

Waiting for review

PR #415

Waiting for review

PR #418

Waiting for review

PR #421

Waiting for review

PR #424

Waiting for review

One pull request, two AIs

Tom's ticket: add a "pay with saved card" button.

AI THAT WRITES · CLAUDE CODE

Sees the whole session

- The ticket and the chat with Tom
- The repo's instruction files
- Provider docs: "retries can charge twice"
- ✓ Writes it without retries

AI THAT REVIEWS · COPILOT

Sees the diff and the repo files

"Consider extracting this into a hook."

"Network calls should have retry logic for failures."

The AI that writes has the whole conversation. The AI that reviews sees the diff.

The human becomes the referee.

Copilot

"Network calls should have retry logic."

Tom

Tags an agent on the PR: "Address the review comments."

Claude

Runs from the PR. Never saw Tom's session. Adds retries.

Copilot

All comments resolved. ✓

Sara

Approved. Twelve PRs in her queue.

THREE MONTHS LATER · SOME CUSTOMERS CHARGED TWICE

“Who decided we retry payments?”

Copilot suggested it.

Claude wrote it.

Tom asked for it.

Sara approved it.

Nobody decided.

THE NEW DRIFT

Every AI going its own way.

Context drift

The AI that writes has the conversation. The AI that reviews has the diff.

Workflow drift

Some check every change. Some say "address the comments".

Review drift

The check itself is now an AI, with its own idea of good.

The costs show up later.

AI review becomes noise

People learn to ignore it.

Decisions nobody made

Everyone approved it. Nobody owns it.

Duplicated effort

Everyone tuning their own AI, alone.

Lost shared understanding

"Why are the retries there?"

Tokens are cheap. Rework isn't.

But a failure in production costs the most: your customers' trust.

AND IT GETS WORSE

The drift **feeds itself.**

Every decision nobody made becomes a pattern everyone copies.

Standardize everything?

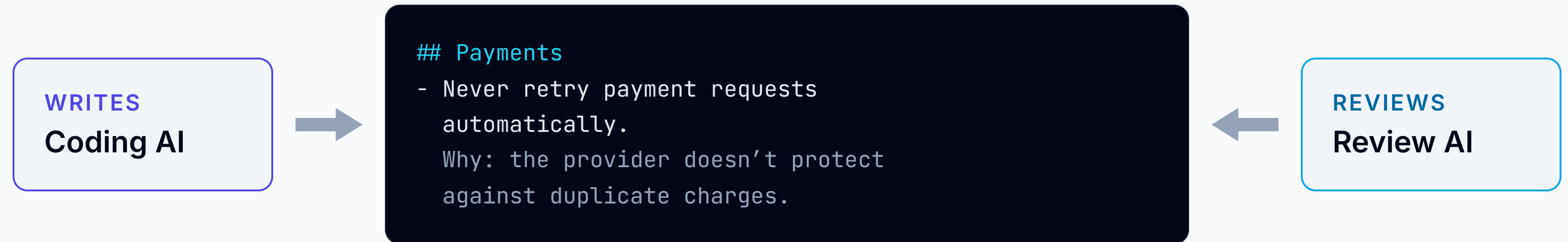
Even with one tool, the AI that reviews would still only see the diff.

SO HERE'S THE IDEA

Align the AIs, not the workflows.

Standardize what every AI knows, and where humans stay in charge.

One context for every AI



Instruction files used to be documentation.
Now they're your quality control.

Who owns the context? **Not just engineering.**

PRODUCT

Business rules and the why

"Payments must never double-charge."

ENGINEERING

Conventions and architecture

"All API calls go through apiClient."

SECURITY · PLATFORM

Guardrails

"No personal data in logs."

CODEOWNERS

`.github/instructions/payments.instructions.md` `@payments-pm` `@tech-lead`

EVALUATION

Humans decide where AI can review

AI review is enough

- Copy changes
- Styling
- Internal tools

A human always reviews

- Payments
- Authentication
- Personal data

```
# CODEOWNERS
/src/payments/ @acme/payments-team
/src/auth/ @acme/identity-team
```

+ "Require review from Code Owners" on main. An AI can't be a code owner.

REVIEW DECISIONS, NOT DIFFS

Humans can't read every line. They can read every decision.

Not “what changed”, but “what was decided, and why”. Updated after every review.

```
## Decisions made
- Added a “pay with saved card” button
- Added automatic retries to payment calls
  (suggested in review)
```

The payments owner spots it in five seconds. And the AI reviewer gets the context too.

KNOWLEDGE · CLOSE THE LOOP

Fix the context, not just the code.

Every incident and every good review catch ends with a change to the shared context.

What we leave free

The tool

The model

The prompts

Agents or chat

That's where the good new ideas come from.

What you can do on Monday

1

Check what your AI reviewer reads. Give it the same context as your coding AI.

2

With your product manager, write down your high-risk zones. There, a human always reviews.

3

Add a “Decisions made” section to your PR template.

No budget. No new tool. Just someone to start.

Same ticket, today

- ✓ The payment rule lives in the repo, with its reason. Both AIs read it.
- ✓ The PR says what was decided, and why.
- ✓ Payments is a high-risk zone, so a payments owner reviews the decisions.

Different tools. Same direction.

For years, we've focused on helping developers work consistently with each other.

The next challenge is helping their AIs work consistently together.

One team.
Many AIs.

Thank you · Pablo Ibanez · pabloibanez.com