skillbench

How Engineers Use GenAI - Turn by Turn, at Bank Scale

An early look at bank-grade telemetry for GenAI-assisted engineering—what's instrumented now, what we're seeing, and what's next.

Matt Beane
CEO, Cofounder, SkillBench
Associate Professor, UC Santa Barbara

Thank you, Gene Kim!

- Came to the community with a study of LLM use in complex, professional tasks in progress
- Gene made introductions allowing for two follow-on studies, early peek today!
- One, qualitative, interview-based: on genAI in SWE/Devops
- One focused on “rocket science” (CTO, world-leading consumer goods): **turn-by-turn telemetry**

← Back to blog

The death of the junior developer

Steve Yegge June 24, 2024

Warning: This blog post is somewhat speculative; the sky might not be falling. But my spidey-sense is definitely tingling. The way we are all doing our jobs in software is changing, potentially in big ways. So let's think of this as a thought exercise.

With that disclaimer, we're off!

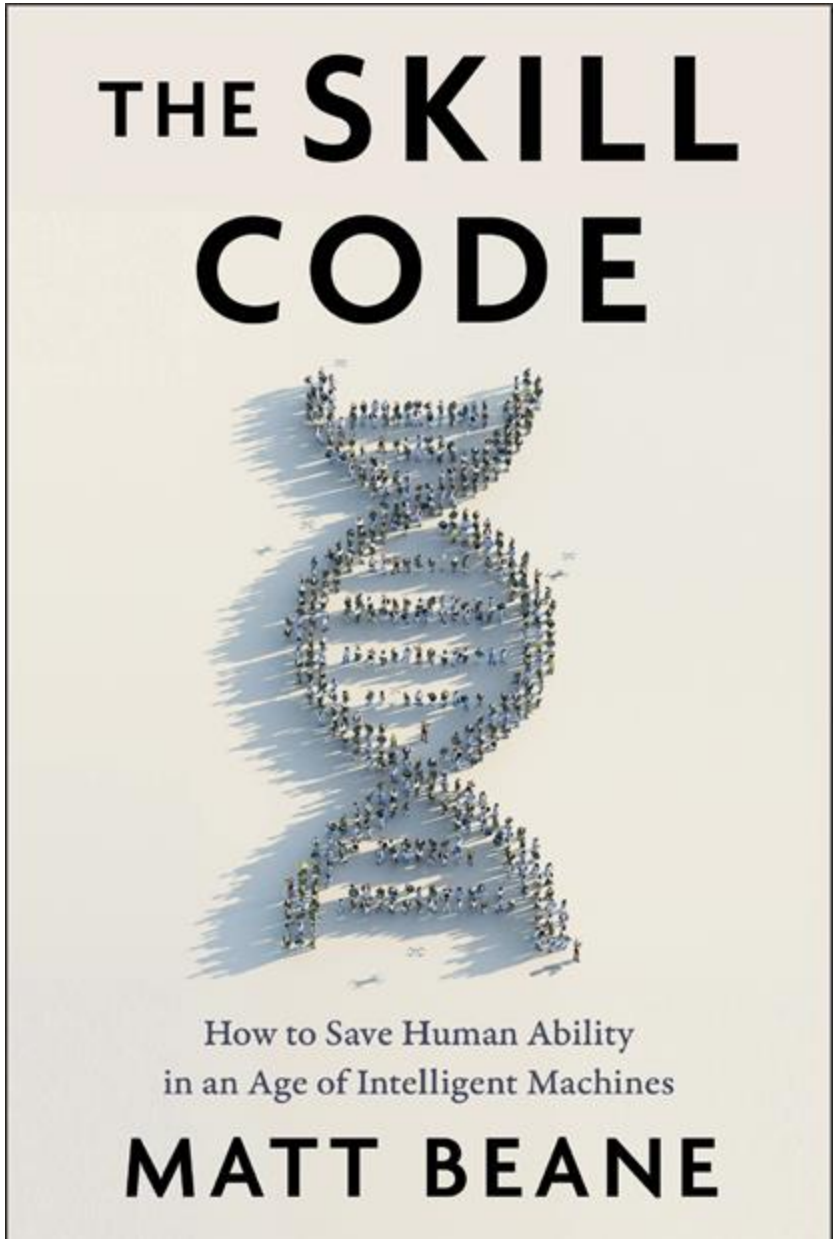
I have been chatting with a bunch of both junior and senior developer-type folk at different companies lately. By cross-barfticulating these conversations I've picked up on an emerging new pattern, which is that like 80% of you are way smarter than me. My goodness. What have I been doing?

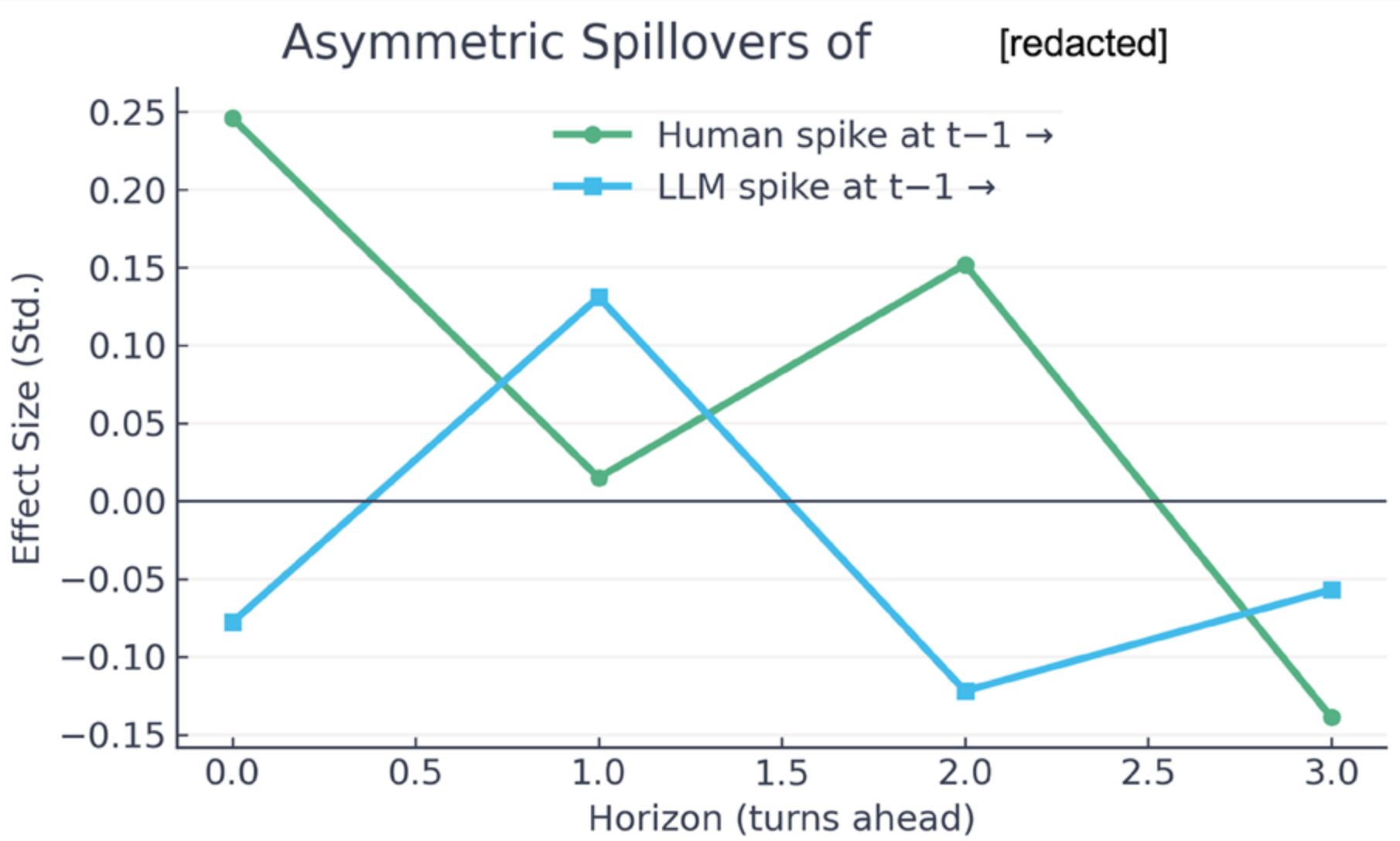
But also another pattern, which is that a lot of people picked a bad year to be a junior developer. A whole lot of people. I wouldn't want to be just getting started in the industry today.

Not just the computer industry. Any industry. It's a bad year to be a junior anything.

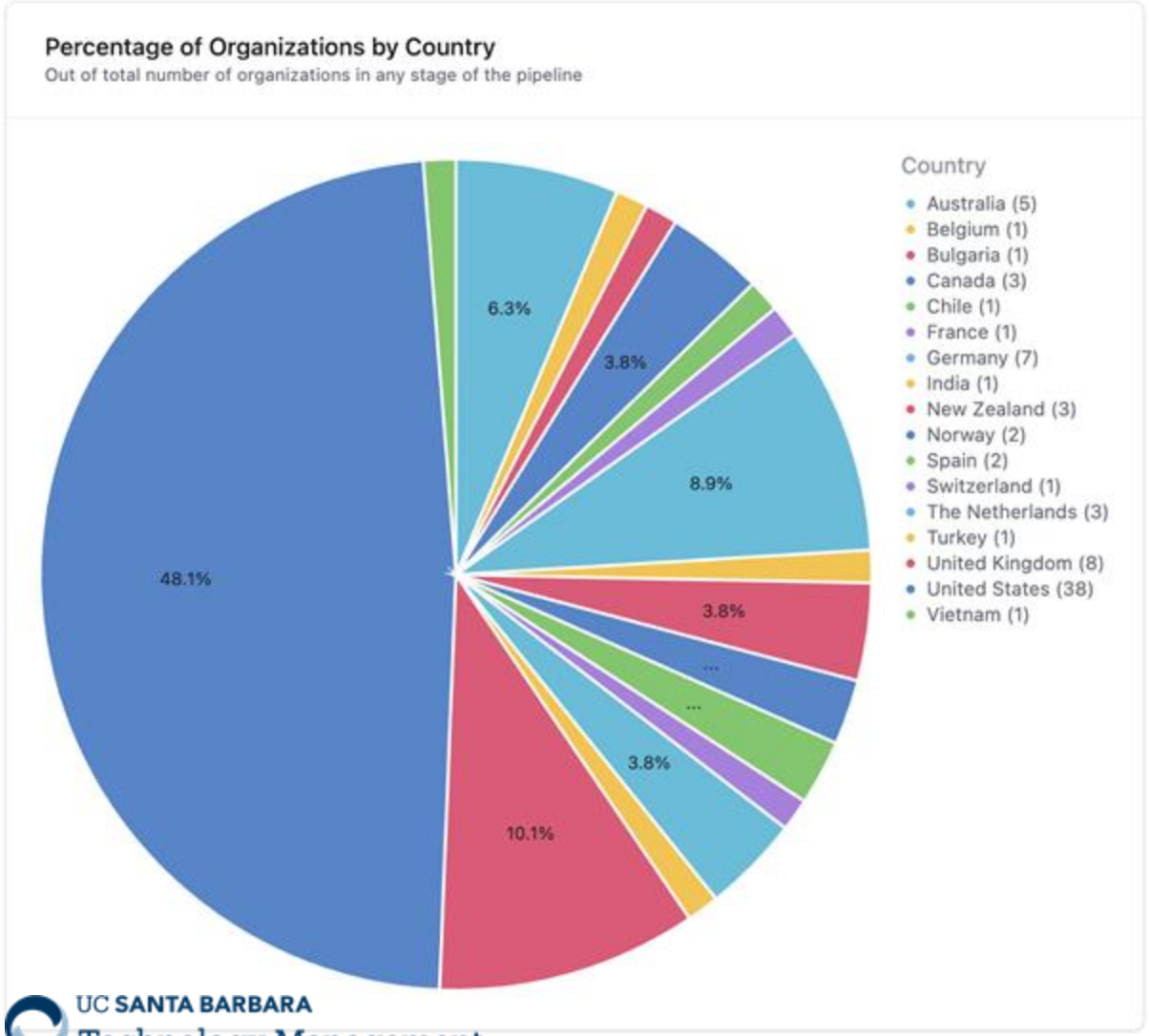
My wife Linh and I are huge horror-movie buffs. So, to reflect the somber theme of today's rant, I will use horror movie titles to introduce our sections.

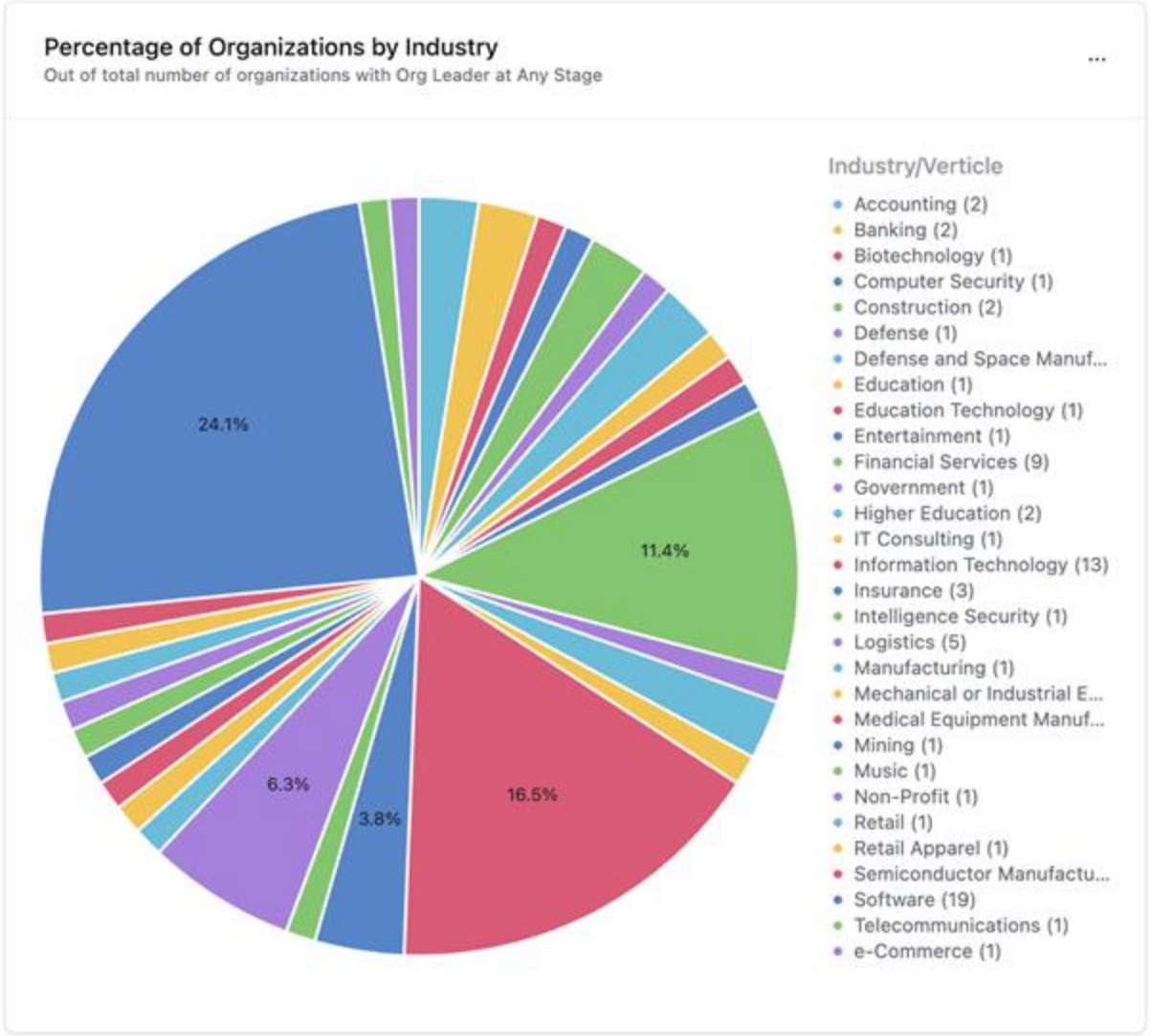
Let's start by talking about what's happening to junior developers.




Asymmetric Spillovers of [redacted]
Human spike at t−1 →
LLM spike at t−1 →
Effect Size (Std.)
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
−0.10
−0.15
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Horizon (turns ahead)

Interview Study: Diverse, Global Sample





Finding 1: Boards, managers, devs want ROI

100%

Finding 2: ROI proof is a challenge

Established Metrics

Code Development

- Breadth of usage: GenAI used in tasks
- Depth of usage
- Tokens per day
- Tokens per task

Code Review

- AI code quality

Team Level

- Team's breadth and depth of usage

Product Level

- Product KPIs over AI features

Test/Ideal Metrics

Code Development

- Use quality: Story points/million tokens

AI-metered DevOps

- Anomaly detection/resolution via log readers

AI Code Review

- Syntax quality (Errors after review)
- Logic quality (Failures after review)

Team Level

- Development Cycle Speed

Product Level

- Portion of customer journey built via AI

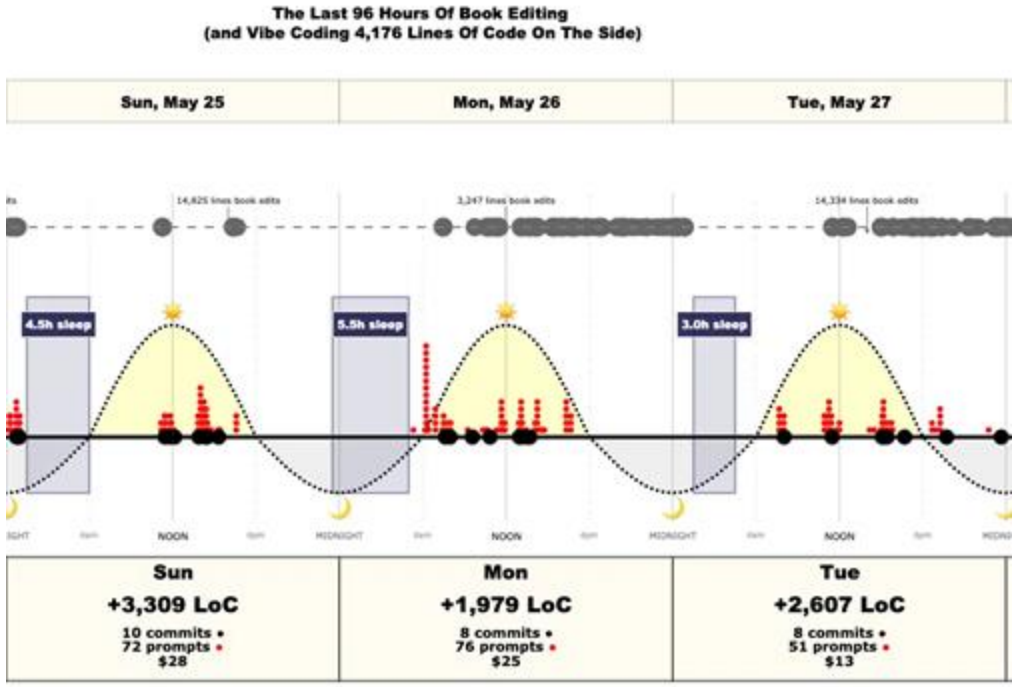
Turn-by-Turn Telemetry: New Yardstick

ROI lives or dies in the sequence of small choices we make with AI

prompt → accept → edit → test → review

Turn-by-turn telemetry enables linking AI use to:

- quantifiable ROI
- productivity impact
- skill development



Gene's "Tufte Diagram" from the Vibe Coding book!

https://itrevolution.com/articles/the-last-80-hours-of-editing-the-vibe-coding-book/

Answering the big Q with Commonwealth Bank

How does variation in GenAI use change CBA engineers' work process and outcomes?

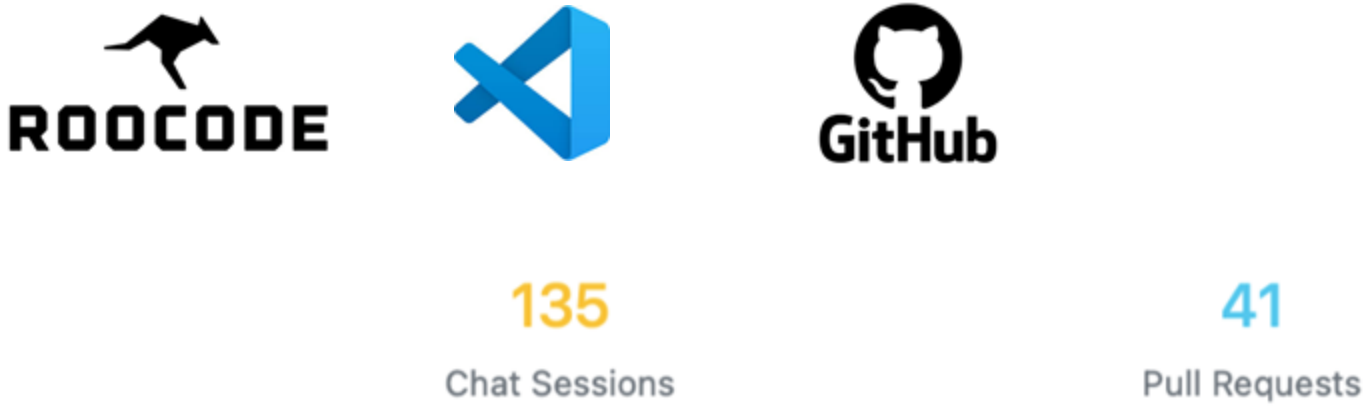
- Productivity, Quality, and Skill as target outcomes
- Turn-by-turn user interaction logs with GenAI
- ~120 SWEs opted in to our preregistered study
- Design is causal: two distinct "nudges" in AI use, four groups
- 2 weeks of data collected so far, will continue until Nov
- Preliminary, suggestive signals today

CIO for Technology, Commonwealth Bank

CommBank Data Collection Overview

Collecting **real-time signals** of developer work from two sources:

- **VSCode Editor** - Capture actual code changes in real time
- **RooCode Agent** - Log interactions between developer and agent
- **GitHub PR** – Outcome signal



Telemetry Signals from VS Code Diff Logs

Document Events

- **Document Open/Close/Save**
- **Text Changes** (with author attribution)
- **File Create/Delete/Rename**
- **Directory Create/Delete**

Workspace Information

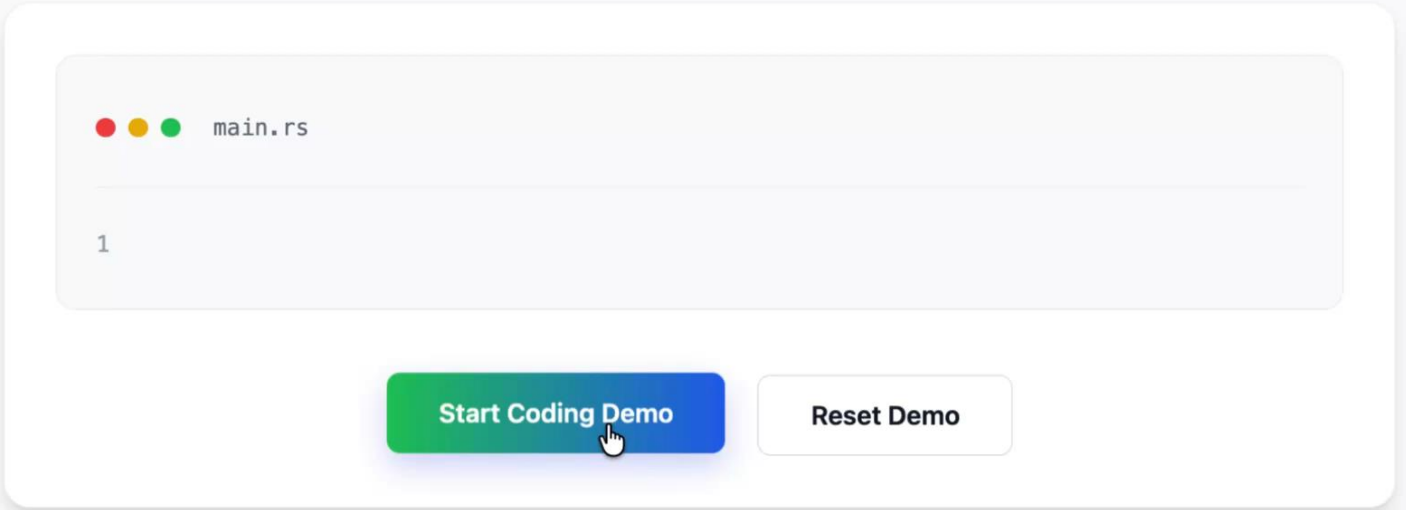
- **File Scan Summary**
- **File Content**

GitHub Information

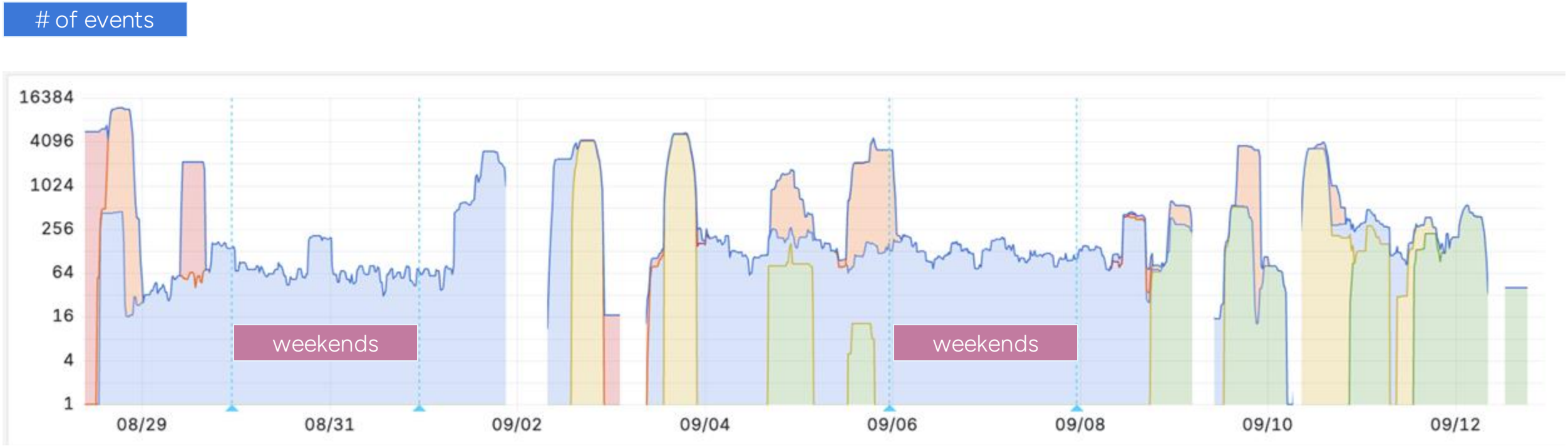
- **Repository Information** (owner/repository)
- **Branch Information**

Secure Code Processing

Watch how we protect your code while preserving essential metadata.
Your intellectual property remains secure throughout our processing pipeline.



Turn-by-Turn Activity Over Time



Each color represents a developer

High-Level AI Interaction Summary

AI Collaboration in Action

- **3,056 AI interactions** across **148 chat sessions**
- Average of **74 prompts per session**
- One developer used **732 prompts in a single session**

Long, Iterative Workflows

- Average session length: **6.5 hours**

Scale of Collaboration

- **72,763 code changes** across **14,131 files**

Time Patterns

- Peak activity: **4 PM**
- Most productive day: **Thursday**

AI is 10x-ing its Devs

- **Total characters:** 28,519,542
  Human: ~3.4M (□ 12%) vs **AI: ~25.1M (□ 88%)**
- **Total lines of code:** 916,573
  Human: ~72.7K (□ 8%) vs **AI: ~843.9K (□ 92%)**
- **AI tokens per session**: 14,697 (21x more than human)

And Devs halfway accept their robot overlords

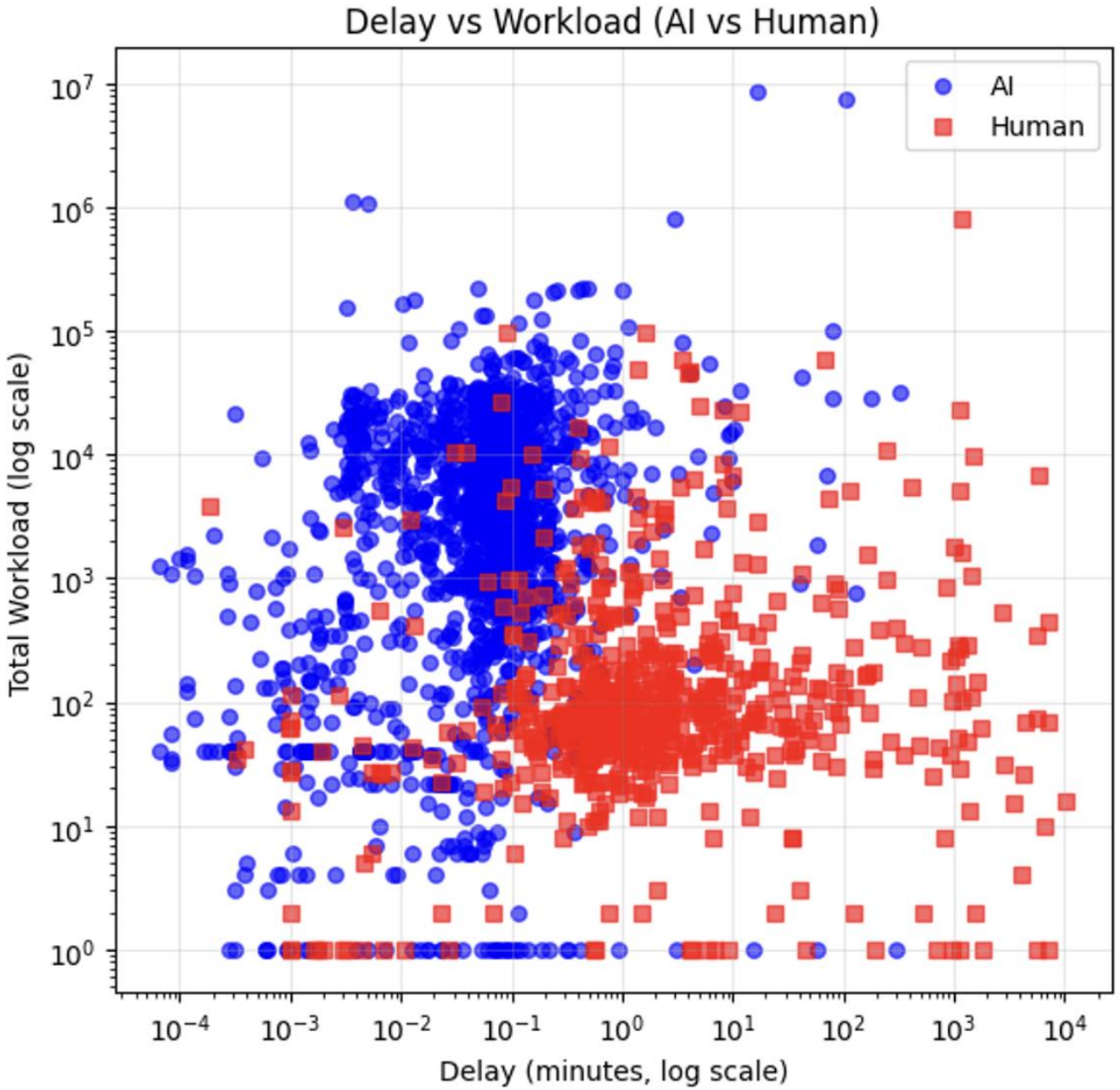
- **AI LOC accepted**: **42.4%** (Nearly half of all code)

AI vs. Dev Rhythm

AI: packs order-of-magnitude more work into short bursts, sprints with little downtime, predictable cadence

Developers: variable pauses, delay is not associated with task size, seconds-scale response appears to be copy-paste

When is delay (in)effective? When is an instant avalanche of code (in)effective?



skillbench

Proposed metric: avg_prompt1_tokens

SAY Text 09:28:15

```
Give me the correct command to use inside the container for my project. the command example is "echo '{"timestamp":"2024-01-15 14:30:00"}' | kafka-console-producer --bootstrap-server localhost:9092 --<NAME> test-input"
```

Vs.

Only one in nine “superusers” regularly started with rich context and intent

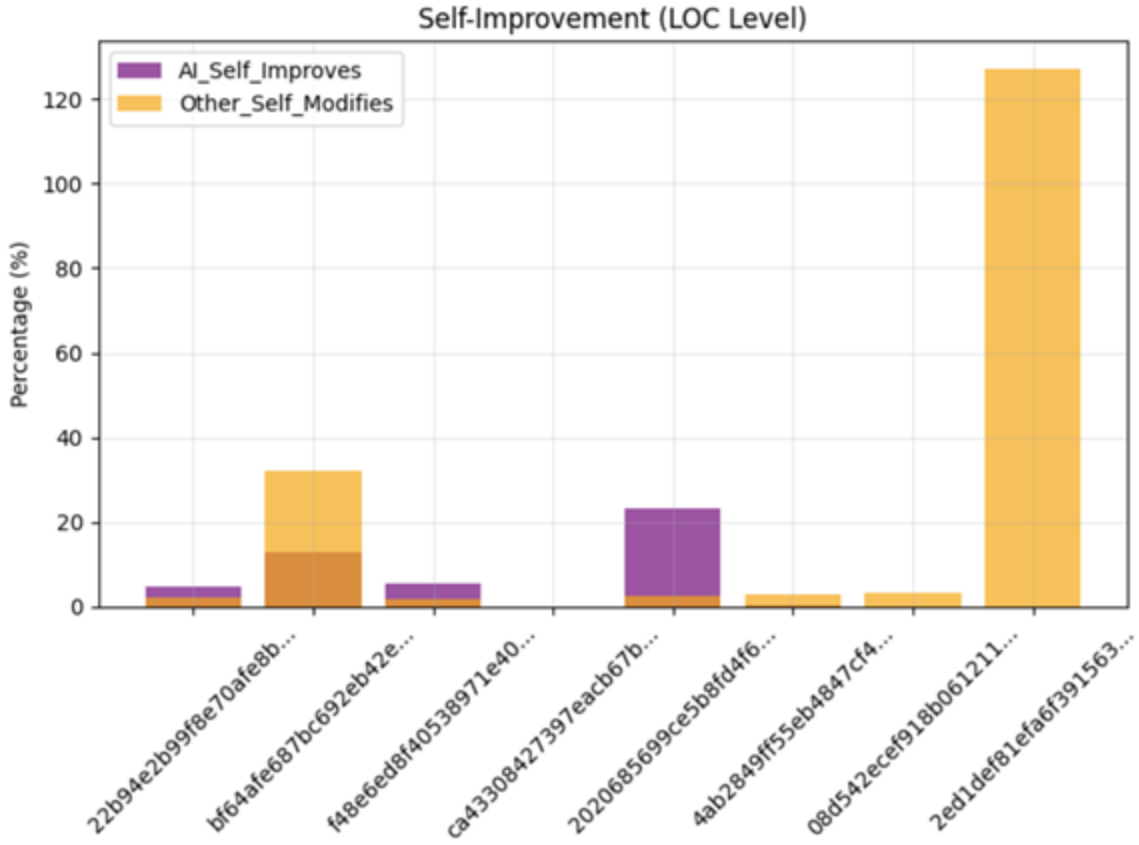
skillbench

Proposed metric: overwrites (trust in AI?)

Overwrite tracking suggests **trust in AI**, **collaboration dynamics**, and even **productivity modes**

LOC-level **overwriting patterns:** AI more often changes human code than the reverse

- AI overwrites Human: 3.56%
- Human overwrites AI: 0.96%
- AI self-overwrites: 5.81%
- Human self-overwrites: 21.51%



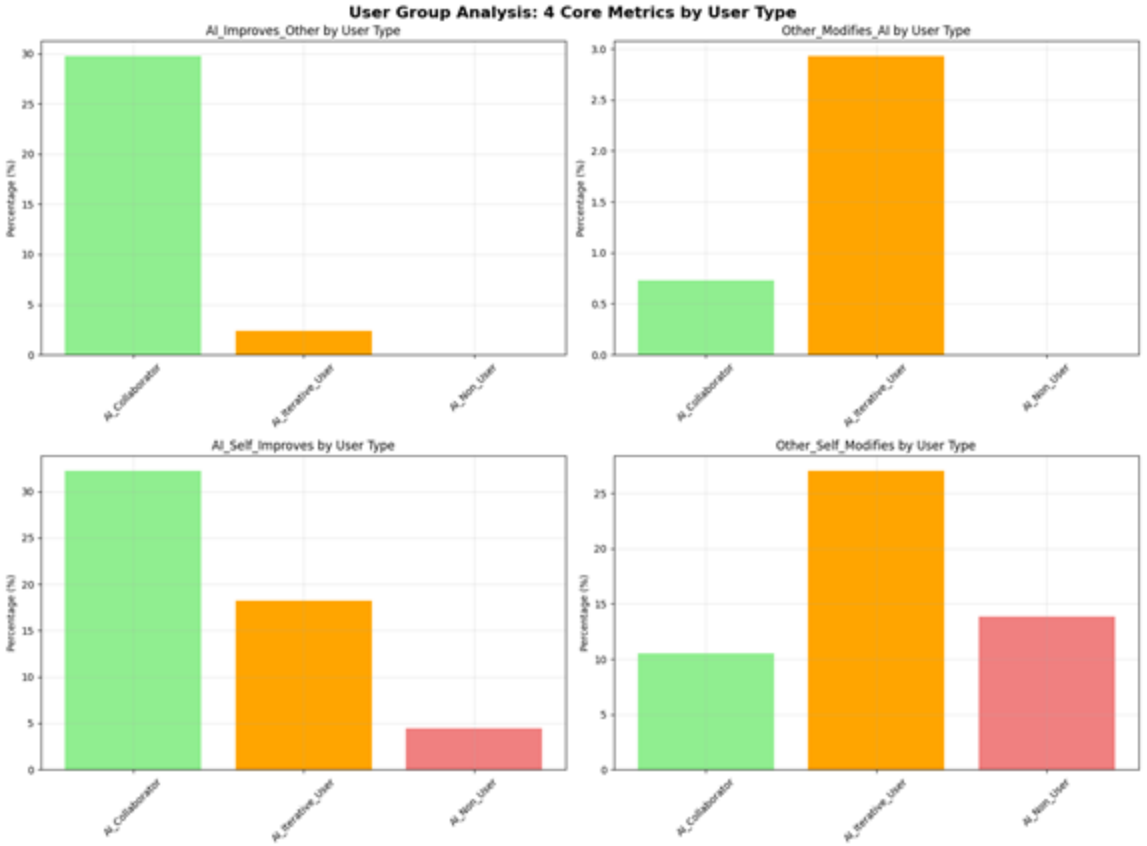
And: trust varies by persona?

Three data-driven personas of AI usage from overwrite tracking

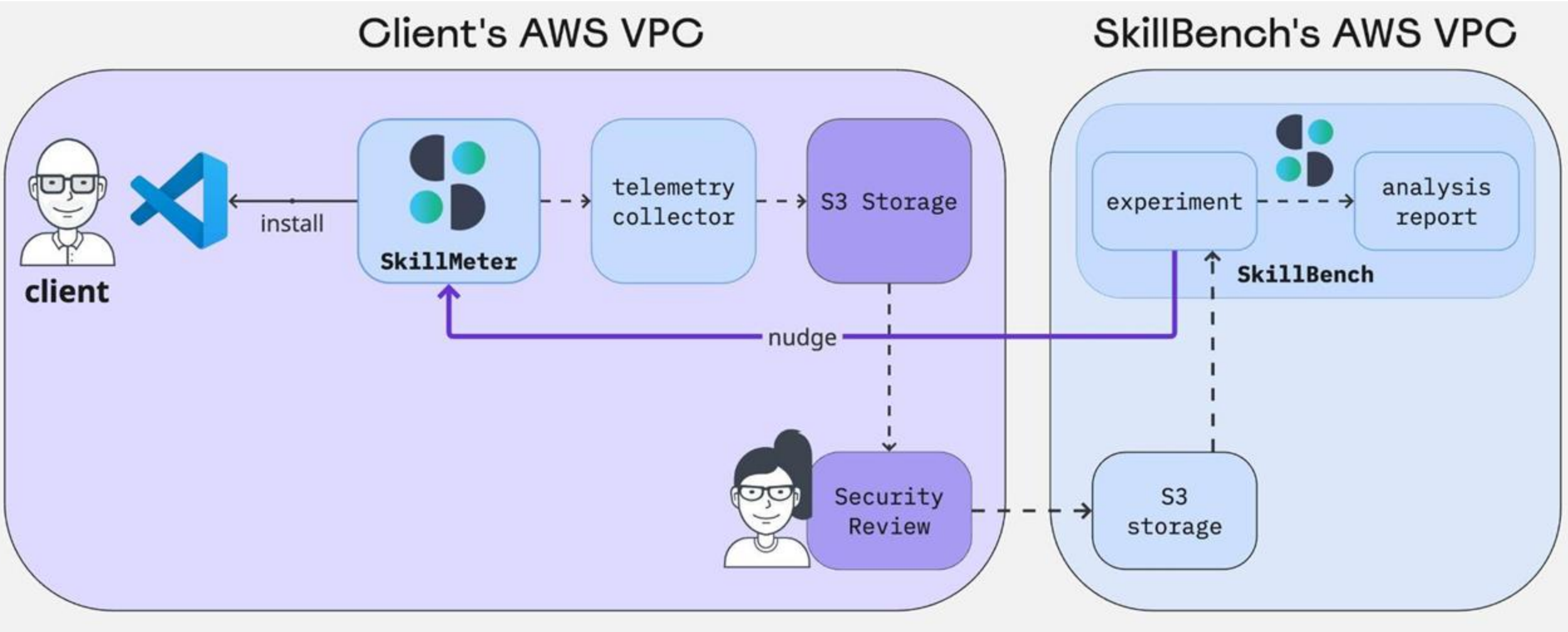
Integrators: High acceptance rate of AI code, AI improves human code, high trust

Iterator: Frequent refinements, human-driven iteration, AI as a sandbox

Keyboardist: Little/No AI usage



Security-First Approach to Data Collection



Early Lessons for Engineering Leaders

- Measure sequences and traces, not just installs. Adoption dashboards miss much of the real story.
- AI and human work cadences differ. AI sprints nonstop; humans are more variable, pause often. Unclear when this is beneficial vs. not.
- Delay seems to be a deep/rich signal. Instant acceptance may mean copying; pauses may mean learning and synthesis.
- Avg_prompt1_tokens seems to matter quite a bit…
- Trust shows up in overwrites. Who edits whose code tells you about collaboration dynamics.

Help needed: take our last pre-launch slot!

We can onboard one additional enterprise partner in the next ~60 days.

Interested? Email **matt@skillbench.com** with subject: “ELTS - Telemetry”

Ps: We’re hiring! If you or someone you know is interested in building with us, please reach out.