Git-First Agent Workflows and Harder Test Prompts Take the Lead

Coding Agents Alpha Tracker

2026-03-22

Git-First Agent Workflows and Harder Test Prompts Take the Lead

By Coding Agents Alpha Tracker • March 22, 2026

The sharpest signal today came from practitioners tightening the loop around coding agents with tests, Git context, and clearer ownership. Also inside: Claude Code web's repo limit, Codex vs. Claude commit attribution, and the clips worth your time.

TOP SIGNAL

The strongest practical signal today: **agent performance is still mostly a scaffolding problem**. Simon Willison says tests, docs, CI/CD, and clean code make agents work better—and his own loop starts with `uv run pytest`; Salvatore Sanfilippo says generic "write tests" prompts miss the hard stuff, and recommends explicitly asking for edge cases, fragile implementation details, and random testing against a simpler reference implementation [1, 2]. Willison's follow-on warning matters just as much: **code review** is now the bottleneck, while **cognitive debt** remains unsolved [1].

TOOLS & MODELS

- **Claude Code for web — current repo-auth ceiling:** Simon says one session can't check out two private repos at once because Git operations go through a local proxy that only authenticates the repo attached to the session. He also says the docs don't mention this [3, 4].
- **Claude Code vs Codex — commit metadata means adoption signals can lie:** Claude Code auto-adds itself as a co-author on every commit; Codex doesn't. OpenAI engineer Tibo Sottiaux says Codex is designed so the user remains the owner and accountable party, even though that makes repo-level usage harder to observe [5, 6].

"it exists to help you and it's important that you remain the owner and accountable for your work without AI taking credit." [6]

- **T3 Code vs Claude Code CLI — creator-posted RAM snapshot:** Theo says T3 Code used **350.9 MB** vs **635.5 MB** for Claude Code CLI in his screenshot, and framed that as roughly **2x** better efficiency [7, 8].
- **Routing pattern worth copying:** Matthew Berman describes a 3-tier stack—frontier models for exploratory work, Sonnet-class models for most execution, and local/fine-tuned models once a narrow workflow is ready for production. His own example was using Opus for front-end/HTML work; Jaden Clark described using a cheaper/default model for small personal tools where speed and cost matter more than max capability [9].

WORKFLOWS & TRICKS

- **Bootstrap a session in 3 moves: (1)** run `uv run pytest`, **(2)** ask for "recent changes" or "last three commits" so the agent runs `git log`, **(3)** only then split into **2-3 parallel sessions** [1, 10].
- **Use Git as an agent power tool, not just a backup:** Ask for `git status` when the repo is messy—Willison says he uses that prompt surprisingly often—then let the agent work through conflicts with tests. For archaeology, have it search the reflog or other branches for lost code, or run `git bisect`; for cleanup, ask it to rewrite history with `git reset --soft HEAD~1`, split/combine commits, or extract a library into a new repo while preserving history [10].
- **Ask for adversarial tests:** Tell the model to stress limit conditions and fragile implementation details, and to use random testing plus a simpler in-test reference implementation to check invariants. Sanfilippo says even a small wording change can strongly steer the model, and the resulting tests become guardrails for both AI-written changes and future refactors [2].
- **Assume review is the scarce resource:** Faster generation just moves the pain to review. Willison's warning is blunt: code review is now the biggest slowdown, and "cognitive debt" is still unsolved [1].

PEOPLE TO WATCH

- **Simon Willison** — published the first draft of Using Git with coding agents. Why it matters: it turns Git from a safety net into an active agent workflow for context loading, debugging, conflict recovery, bisecting, and history rewriting [11, 10].
- **Salvatore Sanfilippo** — Redis creator; today's high-signal contribution was a prompt pattern for stronger tests that targets brittle implementation details instead of shallow happy-path coverage [2].
- **Tibo Sottiaux** — useful because he's surfacing product philosophy from inside Codex: ownership and accountability over brand visibility in com-

mit history [6].

- **Theo** — worth tracking if you care about coding-agent UX tradeoffs; he keeps posting blunt first-party comparisons while shipping T3 Code [7].

WATCH & LISTEN

- **14:39-17:35 — Hard-test prompting that actually changes model behavior.** Sanfilippo explains why "write tests" is too generic, and shows how to request edge-case stress plus random testing against a simpler reference implementation [2].

I test nel software: le tecniche che uso (14:39)

- **1:13:24-1:16:46 — The sim-to-real warning for local/fine-tuned agents.** Shaw Walters says harness-specific data can improve narrow tasks quickly, but may not transfer back to broader benchmarks and can even narrow the model's capability space [9].

09:15 AM PDT
THE FUTURE LIVE
URE LIVE WITH MATTHEW BERMAN AND NICK WENTZ | GUESTS FROM AUGMENT CODE, NEAR PROTOCOL, MODULAR AI | THE FUTURE LIV

The Future Live | 03.20.26 | Guests from MOTS Podcast, Microsoft, Eliza Labs, and Sentient! (73:24)

PROJECTS & REPOS

- **ELIZA OS** — worth watching for routing and safety questions. Walters describes it as an open-source framework for building agents, games, and applications, with deployments ranging from an **8B quantized** model up through **Sonnet** and **Opus**; he also says security is still the blocker for unsupervised browser + shell agents [9]. Adoption signal: the show introduced it as "the most widely used open source framework for building autonomous agents" [9].
- **Sentient Arena / EVO Skill** — still pre-results, but the setup is concrete: the first arena uses **Office QA** for enterprise-style reading, calculation, and document analysis, and the first cohort closes in the first week of April. The notable mechanic is multi-proposal skill evolution from eval feedback; the team says that setup currently does much better with **Opus + Claude Code-style** workflows than with open harnesses/open models [9].

Editorial take: today's real edge was not a flashy new model—it was stronger guardrails around the ones we already have: tests first, Git history in context, and clear human ownership of the output [1, 10, 6].

Sources

1. Profiling Hacker News users based on their comments
2. I test nel software: le tecniche che uso
3. X post by @simonw
4. X post by @simonw
5. X post by @Yuchenj_UW
6. X post by @thsottiaux
7. X post by @theo
8. X post by @theo
9. The Future Live | 03.20.26 | Guests from MOTS Podcast, Microsoft, Eliza Labs, and Sentient!
10. Using Git with coding agents
11. X post by @simonw