Default-Author Coding Agents, Harness Tuning, and Agent-Era Git

Coding Agents Alpha Tracker

2026-05-07

Default-Author Coding Agents, Harness Tuning, and Agent-Era Git

By Coding Agents Alpha Tracker • May 7, 2026

Firsthand reports from Boris Cherny, Simon Willison, and Riley Brown show coding agents moving from assistant to default author. The practical edge is shifting to evals, context control, sandboxing, model routing, and the new tooling shipping around agent-speed development.

TOP SIGNAL

The clearest shift today: for power users, the agent is becoming the default code author, not a sidekick. Boris Cherny says Claude Code now writes all the code he used to write by hand and that he often has anywhere from a few to thousands of agents running; Simon Willison says he already trusts Claude Code for bounded production tasks without line-by-line review, and Riley Brown got GPT 5.5 to turn a Firebase web app into desktop + Swift iOS clients in a 41-minute run [1, 2, 3].

The bottleneck has moved up the stack: plans, evals, context hygiene, and review boundaries now matter more than typing speed, and Simon Willison's *normalization of deviance* warning is the right counterweight as trust rises [4, 5, 6, 7, 2].

TRY THIS

- **Plan first, then force a persistent integration loop.** Riley Brown's GPT 5.5 flow: build the first web app, ask the model to draft a plan, then reply with:

  > "Okay please take a deep look at the plan you made… By the time you are done I want to be able to use all three of these

applications together… Don't stop until you are done." [8]

He used that to convert a Firebase-backed web app into desktop + Swift iOS apps; the run lasted 41 minutes and finished with working auth on both apps [3]. Timeless pattern: specify the finished state, not the next step [8].

- **Black-box only the boring path — and require tests + docs.** Simon Willison's current bar for bounded agent work is tasks like: build a JSON API endpoint that runs a SQL query, outputs JSON, and adds automated tests and documentation [2]. He treats that output as a semi-black box until bugs or performance problems show up, then inspects internals — but explicitly warns that repeated success can create *normalization of deviance* and over-trust [2].
- **Start eval-driven development with 5-10 cases, not 1,000.** Harrison Chase says you can begin with five or ten realistic scenarios, define what a good response and bad response look like, and run every prompt/tool change against that set [4]. Clay's production pattern is to use deterministic checks where possible, LLM-as-judge when needed, and then keep expanding the dataset with real production behavior [5].
- **Sandbox agent internet and shell access — especially for local models.** ThePrimeTime's blunt advice: letting agents roam the internet from your local machine is *the easiest way to shoot yourself in the foot*, while Salvatore Sanfilippo says lack of sandboxing is a major concern when giving less-aligned local models agent-style powers [9, 10]. If you need browser automation, the safer pattern is isolated cloud/browser infra rather than full local permissions [9, 10].

WHAT SHIPPED

- **Claude Managed Agents** *(via Simon Willison's live blog of Anthropic's event)*: multi-agent orchestration and Outcomes are in public beta; Dreaming is a research preview. Outcomes let you define success criteria so Claude can iterate toward them, and Dreaming inspects past sessions to create new memories such as `descent-playbook.md`. Managed Agents overview [11].
- **Claude Code's surface area expanded**: Anthropic showed Code Review, CI auto-fix, Remote Agents, and CLI/IDE/Desktop surfaces built on the Claude Agent SDK; Simon notes Code Review is already used by every team at Anthropic [11].
- **Claude Code limits moved immediately**: 5-hour limits doubled for Pro, Max, Team, and seat-based Enterprise; peak-hour reductions were removed for Pro/Max; Opus API limits were raised. Boris Cherny also said Anthropic's SpaceX partnership adds 300+ MW of capacity and 220K NVIDIA GPUs within the month. Announcement [12, 13, 14].

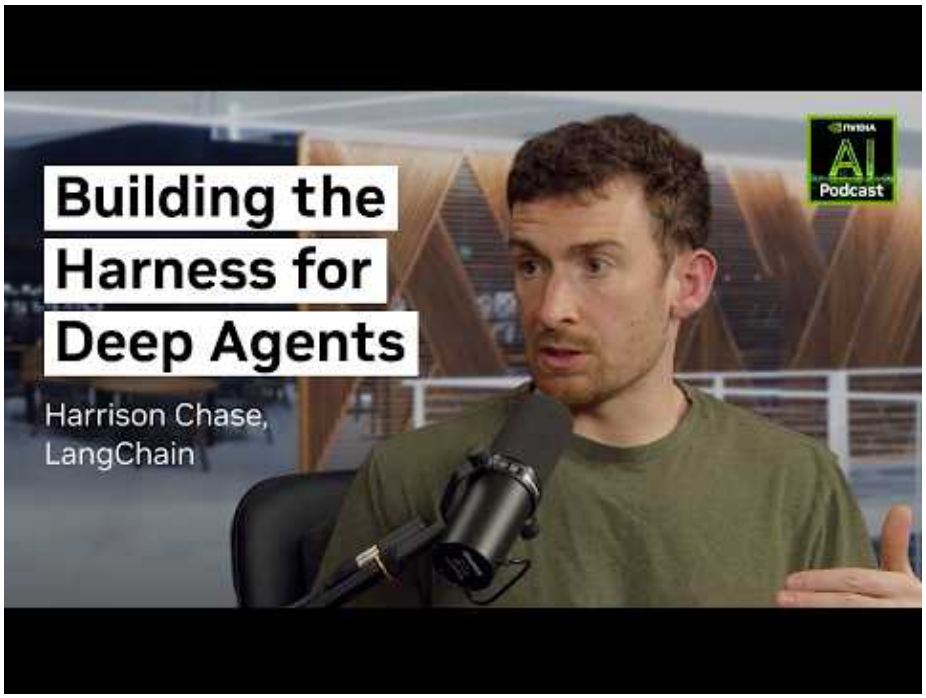
- **Anthropic showed a practical model-routing pattern**: an *advisor strategy* where Opus advises Sonnet on demand. Simon reports Anthropic said one customer, eve, got frontier-model quality at 5x lower cost with this setup [11].
- **LangChain Deep Agents added Harness Profiles**: model-specific overrides for system prompts, tool names/implementations, and middleware. LangChain says its own testing saw 10-20 point gains on a tau2-bench subset versus the default harness; profiles ship for OpenAI, Anthropic, and Google models, with open-weight profiles coming. Tuning blog [6, 15].
- **Cursor 3.3 added context accounting**: you can now see where agent context is going and use the breakdown to debug problems across rules, skills, MCPs, and subagents [7].
- **Cursor's Composer training stack is self-bootstrapping**: Cursor says older Composer generations now set up dev environments for RL training via autoinstall, so newer generations can spend more time on harder tasks. Writeup [16].
- **Agent-era Git infra is getting real**: Theo argues GitHub is straining under agent-generated repo/PR/commit volume; he highlights Pierre / Code Storage for high-throughput repo creation, Entire CLI for preserving the *why* behind agent changes, and says Forgejo/Codeberg is the best open Gen-2 alternative today because its Actions are largely YAML-compatible with GitHub [17].

GO DEEPER

- **10:58-11:23 — Boris Cherny on the new coding loop.** A clean, concise description of the workflow shift: prompt the agent, let it build and test the feature, then approve or request changes [1].

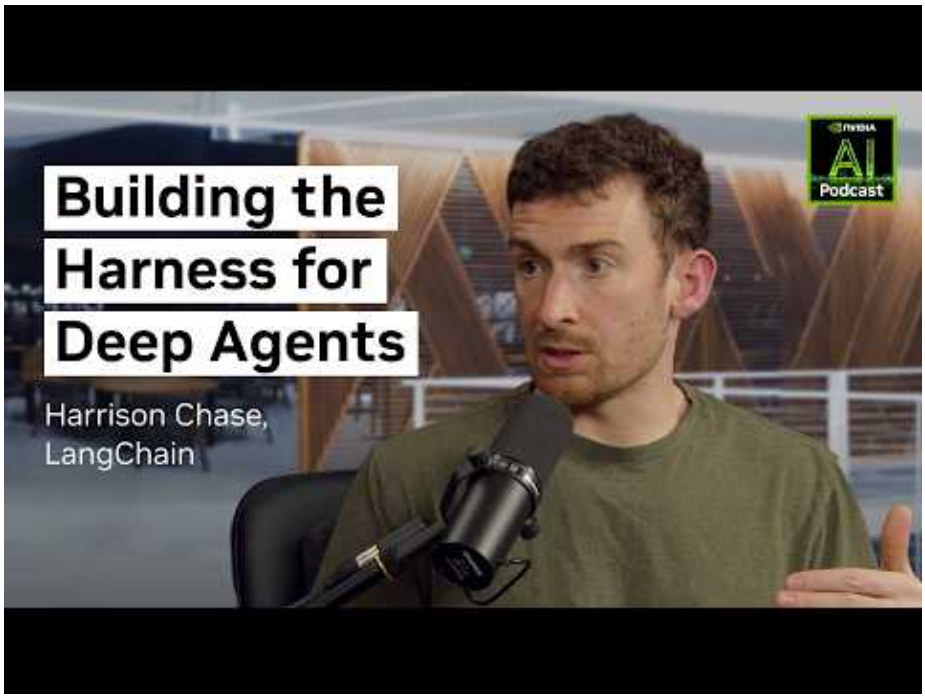
Head of Claude Code on the future of work and productivity (10:57)

- **10:16-11:27 — Harrison Chase on eval-driven development.** The practical takeaway: start with 5-10 scenarios, define good and bad outputs, and let that small eval set govern prompt and tool changes over time [4].



Harrison Chase of LangChain on Deep Agents, LangSmith, and Earning Trust | NVIDIA AI Podcast Ep. 297 (10:15)

- **18:45-22:31 — Harrison Chase on async subagents, proactive agents, memory, and identity.** Useful framing for where coding-agent UX is headed once background runs get long: you talk to the orchestrator while long-running workers do the coding in back [4].



Harrison Chase of LangChain on Deep Agents, LangSmith, and Earning Trust | NVIDIA AI Podcast Ep. 297 (18:45)

- **Read/listen: Simon Willison on where vibe coding bleeds into agentic engineering.** Start with Heavybit Ep. #9, then skim his transcript extract. This is the source set behind his comments on trust, production use, and why the categories are blurring [2, 18].
- **Study Kody.** Kent C. Dodds says the repo hit 160k lines, 428 commits, and 323 PRs since March 18, primarily using cloud agents. It is a live artifact of what agent-heavy OSS development looks like at repo scale [19, 20].
- **Study robobun.** Simon Willison notes Bun's GitHub bot has made more contributions to Bun than Jarred Sumner, which makes this repo worth watching for bot-heavy contribution flow at project scale [21].

Editorial take: getting code out of an agent is no longer the hard part; keeping trust, context, and repo infrastructure intact at agent speed is. [1, 2, 7, 17]

Sources

1. Head of Claude Code on the future of work and productivity
2. Vibe coding and agentic engineering are getting closer than I'd like
3. X post by @rileybrown

4. Harrison Chase of LangChain on Deep Agents, LangSmith, and Earning Trust | NVIDIA AI Podcast Ep. 297
5. How Clay manages 300M agent runs a month with LangSmith
6. X post by @masondrxy
7. X post by @cursor_ai
8. X post by @rileybrown
9. Ticking Timebomb in Mac OS
10. Vlog sugli sviluppi di DS4
11. Live blog: Code w/ Claude 2026
12. X post by @bcherny
13. X post by @bcherny
14. X post by @bcherny
15. X post by @LangChain
16. X post by @cursor_ai
17. What's next?
18. X post by @simonw
19. X post by @kentcdodds
20. X post by @kentcdodds
21. X post by @simonw