Harnesses Matter Now: Cursor Multitasks, GPT-5.5 Spreads, DeepSeek v4 Gets Real

Coding Agents Alpha Tracker

2026-04-25

Harnesses Matter Now: Cursor Multitasks, GPT-5.5 Spreads, DeepSeek v4 Gets Real

By Coding Agents Alpha Tracker • April 25, 2026

The clearest signal today is that coding agents are becoming an orchestration problem as much as a model problem. This brief covers Cursor 3.2's async agent upgrades, practical GPT-5.5 migration tactics, DeepSeek v4's strongest hands-on review yet, and the workflow patterns serious users are actually keeping.

TOP SIGNAL

Harness design is becoming the real differentiator. GPT-5.5 is being singled out for long-running code/data/tool work and natural job monitoring, Cursor 3.2 is shipping async subagents and worktrees for parallel background execution, and Salvatore Sanfilippo shows DeepSeek v4 can be dropped into CloudCode with an endpoint swap while still feeling close to recent closed frontier models in real coding-agent work [1, 2, 3, 4, 5]. The practical takeaway: model quality still matters, but orchestration, migration discipline, and review boundaries are increasingly what decide whether an agent actually ships useful work [6, 7].

TOOLS & MODELS

- **GPT-5.5 keeps spreading across dev surfaces.** Cursor says it is now available there, tops CursorBench at **72.8%**, and is **50% off through May 2** [8]. OpenRouter frames it as SOTA for long-running work across **code, data, and tools** [1]. Romain Huet's API note is the most practical framing: for developers, it gets complex tasks done with **fewer tokens and fewer retries** [9]. Devin's team says it runs longer and more autonomously than any GPT model they have tested [10].
- **Cursor 3.2 = better orchestration, not just better chat.** `/multitask` runs async subagents instead of queueing requests, can

multitask already-queued messages, adds improved worktrees for isolated background tasks across branches, and supports multi-root workspaces for cross-repo sessions [3, 4, 11]. Jediah Katz's recommended pattern: dedicate an async subagent to monitor a background job [12].

- **DeepSeek v4 Pro is the strongest open-weight coding story in today's notes.** In Salvatore Sanfilippo's testing, the **1.6T MOE** model with **49B active params** and **1M context** feels aligned with closed frontier models from roughly **3-6 months ago** and is especially competent for software development [5]. His CloudCode setup was simple: redefine endpoints with env vars/shell script, and the test session cost about **$1/hour** in tokens [5]. He also flags the caveat that benchmark gains are outpacing real-world gains, so don't confuse leaderboard movement with proportional productivity jumps [13].
- **DeepSeek v4 Flash is the local angle to watch.** Sanfilippo says the smaller Flash variant is viable for local inference on a **512GB Mac Studio**, while Pro output pricing was quoted at about **$3.48/M output tokens** and Flash is cheaper [5]. His warning: local coding-agent stability depends heavily on sampling settings, or smaller models can get stuck in repetition loops [13].
- **Current practitioner stack rankings are moving fast.** Mckay Wrigley says his coding split flipped from **80/20 Claude/GPT** to **80/20 GPT/Claude** in under three months, and if he could keep one model for engineering right now it would be **GPT-5.5** [14]. His tool read is blunt: **Codex** and **Claude Code** are T1, **Cursor** is T2 in his current workflow; Codex feels like an engineer, Claude more like a general-purpose coworker [14].
- **Google Cloud's internal harness story matters more than another public benchmark chart.** Thomas Kurian says many engineers use the internal **JetSki** coding harness, that feedback flows directly into Gemini improvement, and Gemini is already used to scan for security issues before senior review and to troubleshoot cloud incidents by exposing tools/APIs to the model [15].

WORKFLOWS & TRICKS

- **When migrating to GPT-5.5, don't treat it like a drop-in.** OpenAI's guidance is to start from the smallest prompt that preserves the product contract, then retune reasoning effort, verbosity, tool descriptions, and output format instead of hauling over your whole old prompt stack [6]. If you want the lazy path, Simon Willison points to the Codex command `openai-docs migrate this project to gpt-5.5`, and Romain Huet explicitly suggests asking Codex to migrate a Responses API integration for you [6, 9].
- **Force a short status update before any tool calls on multi-step work.** OpenAI's prompting guide recommends a 1-2 sentence user-visible update that acknowledges the request and states the first step; Simon

notes Codex already does this, and it makes long runs feel much less like the model crashed [6].

- **Cursor's best new pattern: spawn a watcher, not more queue.** Use `/multitask` to create an async subagent, let it monitor a background job, and keep the main thread moving; queued messages can also be converted into multitasked work instead of waiting for the current run to finish [12, 3].
- **If you evaluate coding agents, steal Sanfilippo's harness.** Give the model a small but real codebase, a hard line-count budget, a non-trivial test suite, benchmark programs, and explicit anti-benchmaxing rules; then only count wins if speed improves with no regressions [5, 13]. His optimization hints—dual-ported objects, stack-machine expressions, fixed local-variable slots—show how to give strong priors without hand-writing the patch [13].
- **Human-in-the-loop still wins at the PR boundary.** Kent C. Dodds says he can let agents work through personal-but-complex software mostly on their own, then review the PRs when they are done [7]. Google Cloud is running the same shape at org scale: model-first inspection, human peer review retained, and exploration of separate supervisor models for review [15].
- **Measure output like Google does: functions shipped, not lines of code.** Kurian's point is simple: senior engineers write more compact code, so LoC is a bad productivity metric in an agent-heavy workflow [15].

PEOPLE TO WATCH

- **Salvatore Sanfilippo** — one of the few people doing repeated, same-task comparisons across frontier and open-weight models instead of screenshot benchmarks. His DeepSeek v4 tests and local-inference notes are useful because they include both wins and caveats [5, 13].
- **Jediah Katz** — high-signal on agent UX right now. The useful detail today was not just that GPT-5.5 is strong, but that it is strong specifically at multitasking and monitoring long-running work—and Cursor is shipping around that behavior [12, 2].
- **Geoffrey Huntley** — worth tracking for timeless agent patterns. His Ralph Wiggum memory-management loop is now built into Claude, Cursor, and Copilot, and his bigger point is that deliberate practice still separates casual use from real leverage [16].
- **Kent C. Dodds** — clean articulation of the end-state workflow: autonomous execution first, human review second [7].
- **zeeg + ThePrimeagen** — useful anti-hype filter.

> "The state of the art is still 'can we even one shot a production quality patch that we wont regret later', and its rarer than you'd expect based on discourse." [17]

Primeagen says he likes this framing not because he is anti-AI, but because

obsessive prompt-chasing can wreck sleep, relationships, and life balance [18].

WATCH & LISTEN

- **07:47-12:37 — Salvatore Sanfilippo's coding-agent benchmark design (Italian).** Best technical segment of the day if you care about evaluation quality: a tiny interpreter, 70 tests, hard code-size limits, explicit speed targets, and anti-benchmaxing constraints [13].

Alcuni test sui recenti modelli alternativi (7:47)

- **15:13-17:17 — Why local coding agents still loop and degrade (Italian).** Useful reality check on OMLX and local inference: fast runtimes are not enough if repetition penalties and sampling are off [13].

Alcuni test sui recenti modelli alternativi (15:13)

- **00:05-03:12 — Riley Brown on Codex + Remotion.** Good hands-on walkthrough of why built-in plugins matter: one interface for prompts, code generation, and rendered artifacts, with a very copyable project setup [19].

Codex Just Replaced 1,000 Hours of Video Editing Tutorials (0:05)

PROJECTS & REPOS

- **Gondolin** — sandbox project supporting **QEMU**, **krun**, and **WASM** on a branch [20]. The interesting bet: its builders picked **QEMU** over Firecracker because they think future agents need "the computer they'll actually need," not just a thin function runtime [21].
- **Ralph Wiggum loop** — not new, still relevant. Huntley describes it as the memory-management technique now built into **Claude, Cursor, and Copilot**, and says it spread through YC startups in early 2024 [16]. The core pattern is still simple: keep appending working memory to an array and resend it to a stateless API in a loop [16].
- **OMLX** — MLX-based local inference tooling for Mac worth watching if you want local agent runs with larger open-weight models [13]. The caveat is the point: speed is nice, but stable coding-agent behavior depends on tuned repetition penalties and sampling [13].

Editorial take: the edge is shifting from "who writes the prettiest diff" to "who can keep a long-running agent on the rails, visible to the user, and reviewable at the end." [1, 3, 6, 7]

Sources

1. X post by @OpenRouter
2. X post by @jediahkatz
3. X post by @cursor_ai
4. X post by @cursor_ai
5. DeepSeek v4
6. GPT-5.5 prompting guide
7. X post by @kentcdodds
8. X post by @cursor_ai
9. X post by @romainhuet
10. X post by @cognition
11. X post by @cursor_ai
12. X post by @jediahkatz
13. Alcuni test sui recenti modelli alternativi
14. X post by @mckaywrigley
15. Google Cloud CEO: Anthropic, TPUs, Mythos, NVIDIA and more
16. Geoffrey Huntley - Software Development Now Costs Less Than Minimum Wage
17. X post by @zeeg
18. X post by @ThePrimeagen
19. Codex Just Replaced 1,000 Hours of Video Editing Tutorials
20. X post by @mitsuhiko
21. X post by @utpalnadiger