Agentic Engineering Takes Over the Coding Agent Playbook

Coding Agents Alpha Tracker

2026-05-22

Agentic Engineering Takes Over the Coding Agent Playbook

By Coding Agents Alpha Tracker • May 22, 2026

Karpathy's "agentic engineering" framing was the clearest signal today: the leverage is moving from raw codegen to spec-writing, context design, execution environments, and cleanup discipline. This brief covers copyable workflows, new Codex and LangChain releases, Composer 2.5 economics, and the clips/repos worth studying.

TOP SIGNAL

- **Karpathy gave the clearest framing of the day:** vibe coding now raises the floor, but *agentic engineering* is the discipline of keeping the old quality bar intact while agents handle recall-heavy implementation and boilerplate [1]. Addy Osmani and swyx showed the practical version of the same shift: push your effort up-stack into precise specs, adversarial review, and repo hygiene—not more manual typing [2, 3].

TRY THIS

- **Use prompts as the install script.** Karpathy's OpenClaw example is simple: replace a branching shell script with a copy-paste instruction block, hand it to the agent, and let it inspect the environment, adapt the setup, debug failures, and finish the install [1]. This works best on tasks where the result is easy to verify [1].
- **End every session with a learning pass, then a hostile review pass.** Add an instruction to your agent markdown/config like: `At the end of this session, codify the learnings or ask me what was new today.` [2] Before you push, ask: `What did we miss?`, `What objections exist?`, `What parts are unclear?` and, if needed, `don't`

`be nice` [2]. Addy Osmani's point is mutual amplification: both you and the agent should get better every day [2].

- **Split planning from execution.** Matthew Berman's recommended routing: use frontier models such as Opus 4.7 / GPT 5.5 for the upfront plan, then hand actual code writing and iteration to a cheaper workhorse model like Composer 2.5 [4]. His economics case is concrete: Composer 2.5 sits roughly 1.5 points off frontier on Cursor Bench at about $0.55/task, with pricing at $0.50/M input and $2.50/M output [4]. For your own evals, Boris Cherny recommends keeping recurring real-world "test cases" and rerunning them across models to spot genuine capability jumps [5].
- **Automate only the verified slice, then widen it.** Berman's OpenClaw lesson was that maintenance overhead can dominate: he says ~95% of his total effort ended up in maintaining the automation as fast-moving software kept breaking it [4]. His safer loop is manual first, then automate one part, confirm it works, then extend the automation step by step [4]. Good rule when the end-to-end agent demo is impressive but brittle [1].

WHAT SHIPPED

- **Codex got a real usability push on Mac + iPhone.** Appshots lets Mac users press `Command-Command` to attach the current app window to a Codex thread, including a screenshot plus full text from the window—even beyond what's visible onscreen; it's live across plans on Mac, with enterprise access coming soon [6]. Remote Computer Use now lets Codex Mobile control Mac apps even when the machine is locked; docs: locked-use docs [7]. The ChatGPT iOS app also added turn-completion push notifications, better reconnection, a more compact conversations UI, `/fork`, and improved diffs with full-file open [8, 9].
- **Codex enterprise controls are showing up too.** Greg Brockman called out token analytics and plugin sharing for business/enterprise users [10].
- **Claude Code is adding token attribution.** The upcoming `/usage` command breaks down token consumption by Skills, Agents, MCPs, and Plugins; CLI first, Desktop next [11].
- **LangChain opened private beta for Managed Deep Agents.** It's model-agnostic managed infra for deep agents, with `deepagents init → deepagents deploy → live endpoint`, a versioned context hub, per-thread sandboxes with persistent filesystem, shell access, file I/O, auth proxy, snapshotting, and a durable execution harness with no Dockerfile or infra glue. Full breakdown: Introducing Managed Deep Agents [12, 13, 14, 15].
- **Cursor filled in the Composer 2.5 stack and pricing.** The model is built on Kimi K 2.5 and trained with 25x more synthetic tasks than Com-

poser 2 [4]. Pricing is $0.50/M input and $2.50/M output, performance is ~64% on Cursor Bench and 63 on Artificial Analysis's Coding Agent Index, and it's only available inside Cursor [4].

- **Daytona is an infra project worth watching if you care about agent sandboxes.** In Latent.Space's interview, CEO Ivan Burazin says Daytona starts one sandbox in ~60ms, can bring up 50,000 concurrent sandboxes in ~75 seconds, and already serves a customer close to 850k sandboxes/day; RL/eval workloads have gone from 0% to about 50% of usage in just a few months [16]. The comparison to EKS/GKE is speed, ergonomics, stateful snapshots, and dynamic resizing rather than generic container orchestration [16].

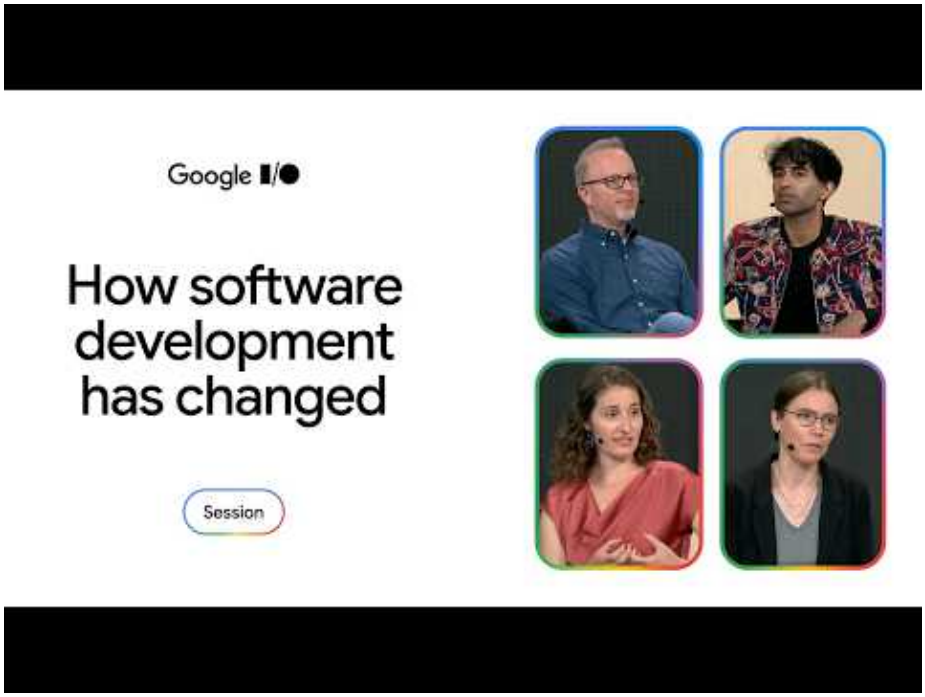
GO DEEPER

- **15:57-17:10 — Karpathy on "vibe coding" vs "agentic engineering."** Best short clip of the day if you want a clean mental model for where the human still matters: spec, taste, security, and fundamentals [1].

[ ] Andrej Karpathy From Vibe Coding to Agentic Engineering (15:57)

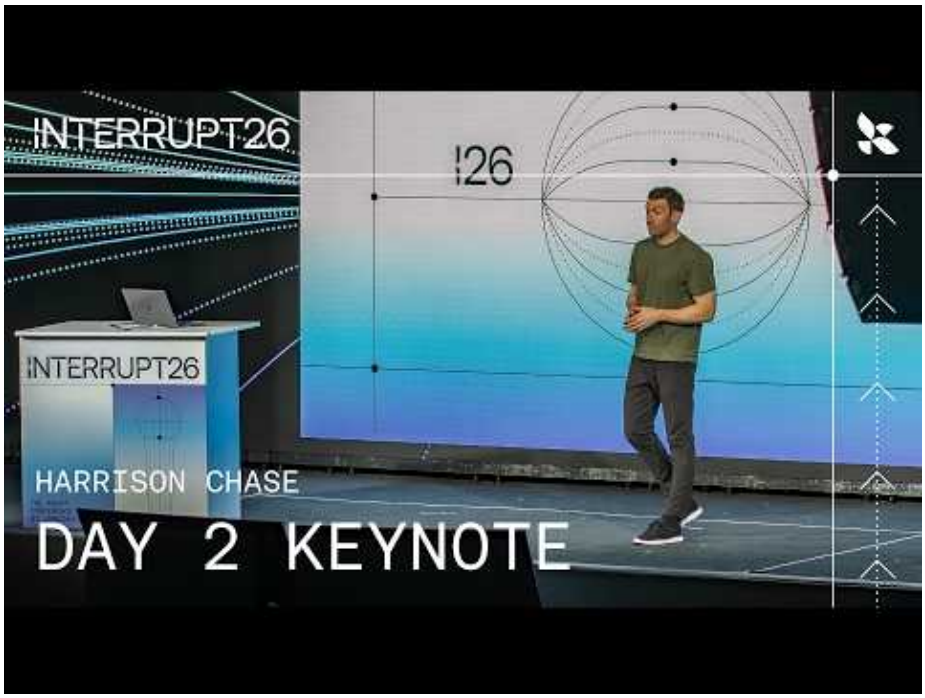
- **27:23-28:40 — Addy Osmani's adversarial mentor pattern.** Good pre-push ritual: ask the model what you missed, what objections exist, and what is still unclear. This is one of the best quality-over-velocity

habits in today's sources [2].



A fireside chat on the evolution of the developer craft (27:23)

- **08:46-09:18 — LangChain on Metaharness.** Worth watching if you still think model choice is the whole game: they cite MIT/Stanford's Metaharness and say they moved from top-30 to top-5 on Terminal Bench 2 by editing the harness only, with no model change [17].



The Future of AI Agents: What Will Interrupt 2027 Look Like? | Interrupt 26 (8:45)

- **16:45 — Daytona on sandbox economics.** If you're deciding whether agent infrastructure belongs on generic Kubernetes or something more stateful, this podcast segment is the most concrete operator-level discussion in today's set: startup time, concurrent scale, snapshot distribution, and why RL/eval workloads look different from "follow-the-sun" agents [16].
- **Repo to study: datasette-agent.** Simon Willison says Claude Code and Codex are both strong at writing plugins against this repo, which makes it a nice small-but-real benchmark if you want to compare coding agents on something more practical than a toy prompt [18]. If you want a local-model loop, he also shared a `uv` one-liner for running it via LM Studio [18].

Editorial take: the durable edge is shifting away from "who can prompt fastest" and toward who can define the spec, shape the context, choose the right execution layer, and keep the cleanup bar high. [1, 2, 17, 19]

Sources

1. [ ] Andrej Karpathy From Vibe Coding to Agentic Engineering

2. A fireside chat on the evolution of the developer craft
3. X post by @swyx
4. Composer 2.5 and I INTERVIEWED THE CEO OF ALPHABET
5. Claude Code Head Boris Cherny: My AI Booked Eight Flights And Five Hotels Autonomously
6. X post by @OpenAIDevs
7. X post by @OpenAIDevs
8. X post by @Dimillian
9. X post by @romainhuet
10. X post by @gdb
11. X post by @bcherny
12. X post by @LangChain
13. X post by @LangChain
14. X post by @LangChain
15. X post by @LangChain
16. Giving Agents Computers — Ivan Burazin, Daytona
17. The Future of AI Agents: What Will Interrupt 2027 Look Like? | Interrupt 26
18. Datasette Agent
19. X post by @ericzakariasson