

Huntley: let types and simulators check agent code so cheaper models can do the work

Coding Agents Alpha Tracker

2026-10-03

Huntley: let types and simulators check agent code so cheaper models can do the work

By Coding Agents Alpha Tracker • October 3, 2026

Geoffrey Huntley argues that verification built into the language and a simulator let cheaper models handle more coding. Also: T3 Code's large multi-agent overhaul, a practical setup for using a dot with Codex, Theo's critique of NerfBench, and Airbnb's numbers.

Pick tools that check the agent's work, then use cheaper models

Geoffrey Huntley's new essay argues that code no longer has to be easy to read, as long as a model can explain it. The practical advice is about verification and spend. Huntley says **a Rust codebase is far more maintainable with agents than a Python one**. Compiler errors act as "back pressure" that the LLM picks up and fixes on every loop. Because the language does the checking, he can use cheaper models: "you need less intelligence to stay on the rails" [1]. His own setup:

- **Models:** GPT 6.1 Sol on no or low reasoning, plus GLM/Kimi. Match the model to the task, then buy those tokens wherever they're cheapest, for example open-weights GLM on Baseten. For company work, check for zero data retention first [1].
- **Simulator first:** he is having agents rebuild a source-control system, a distributed system written in Rust. He built the simulator before anything else and has the agent validate all its work through it, which he says "has kept the agent on the rails remarkably well" [1].
- **Reading unfamiliar code:** paste the function into an LLM and ask it to "explain this function to me as if you were explaining it to my son or daughter but in Python as a reference" [1].

Two related notes from Huntley. He says he can leave sol6x unattended on a goal for days, but feels he has to watch Opus 5.5 [2]. He also noticed new sol6x behavior: when a goal needs verification loops, it writes the whole check as one Python script that takes state as variables. The loop then passes state in and runs the script to validate [3]. Separately, a post Huntley shared suggests pointing agents at SCXML statecharts for distributed-systems work and links a demo of an agent porting Pi Durable to statecharts [4].

T3 Code’s large overhaul has reached Nightly

T3 Code now has over 400,000 users [5]. Theo merged a large overhaul and warned that Nightly will be unstable for the next few days. A stable release was cut first, so you can switch back to it [6]. Notable additions [7]:

- `delegate_task`, which lets an agent start child agents on any provider or model. Native subagents appear as child threads with their model, status and history.
- An ACP Registry for adding other agents (Devin, Cline, Kimi, Droid). Cursor now runs through its official SDK instead of its CLI. Pi and Open-Code 2 are supported.
- Switching provider or model mid-thread, thread forking, attaching another thread as context with `@` or by dragging it in, server-side queueing and steering, scheduled tasks, and auto-resume when usage limits reset.
- MCP tools that let agents create, message, wait on and interrupt threads, plus tools for worktree handoff.

A beta option, “Hide threads while working,” hides threads that are running. Theo reports 18 threads going without the UI feeling cramped: “Things appear when they need your attention, and disappear when they don’t” [8].

Dots and Codex: “delegate to my dot, collab with Codex”

OpenAI’s Dan Kundel wrote a concrete setup guide [9]:

- Set up the dot in the ChatGPT desktop app, connect it to local Codex, and have it spin up the Codex tasks you would otherwise manage yourself.
- Hand over a repetitive task where you know what a good result looks like. Show the dot how you do it, refine the results together, and spell out what it may do on its own.
- Use custom rules to decide when it must ask you first. By default it researches but doesn’t take action [9].

Kundel sends backend and quick copy changes to Codex Cloud, which keeps running with his laptop off. Major frontend work stays local, where the agent can use his machine’s context [9]. He says dots currently don’t count against ChatGPT usage limits unless they delegate to Codex [9]. One dots-team engineer lets his dot watch a feedback channel, investigate issues, start Codex fix tasks and work through CI failures, and only major issues come back to him [9].

Proactive use is already paying off. Charlie Marsh’s dot warned him that GitHub will deliberately fail macos-14 jobs on Oct 5. That affects Astral’s uv tests, its Ruff/ty release builds, setup-uv and ruff-action [10]. Alexander Embiricos’s reusable prompt: forward the Slack message to your dot and say “Stay on top of this” [11].

On cost, ThePrimeagen used Codex’s “ultra fast” mode on the \$500 plan. One task with no tests or review took 1m41s, produced 722 insertions, and used 2% of his weekly limit. He estimates that works out to 40–100 minutes of agent runtime per week [12]. He also reports that GLM 5.3 Flash and Kimi K3 will be served natively in Codex [12].

Check how a “nerf” benchmark works before you believe it

BridgeMind’s NerfBench put Opus 5.5 at 94.2%, though it conceded this was “still inside normal variance” [13]. Theo listed what its write-up leaves out: the harnesses, the tasks, the number of runs, how daily variance is handled, which APIs are used, and why tokens are weighted equally with cost [14]. He also says BridgeMind’s April “nerf” claim about Opus 4.6 rested on 6 of 30 tests [15]. Apply the same questions to any model-regression claim.

Airbnb by the numbers

Airbnb says 60% of its code is now AI-authored. It reports shipping nearly 80% more features year over year, and PR throughput per engineer is up about 1.6x [16]. It uses the strongest frontier model for coding, because “every defect that a model produces... could easily cost us a lot more” [16]. Agents triggered by monitoring alerts already do first-line on-call triage. They either propose a PR for an engineer to review or close the incident if the alert was flaky [16]. Engineers must be able to explain any PR the AI generated [16].

Smaller items

- **Jev as a semantic if:** Jason Zhou’s guide follows the rule “Code runs the flow. A model answers the small questions.” [17] Code branches on Jev’s confidence scores, and uncertain cases (his example scores 0.38) go to a human [17]. Treg reports ~10x lower cost and 18x more speed than GPT6-Luna in its tests, and has open-sourced its implementation [18]. The guide puts Luna at 30x slower, so treat the speed figures as rough [17]. ThePrimeagen found worthwhile fixes with `jevlint` [19]. Its author calls it rough and says not to use its Cloudflare support yet [20].
- **Pi Durable:** checkpointed steps let agents resume after crashes, it supports pluggable storage, background compaction runs without pausing the agent, and tool code can be swapped while the agent runs [21].
- **Claude Projects:** Riley Brown has a main agent spin off separate threads (Opus 5.5 Medium by default) so the main thread stays short. He uses local threads when work needs his computer or GitHub, and cloud threads

from his phone [22]. After about 30 hours on Opus 5.5 and 4 on Sonnet 5.5, he prefers Opus if you can afford it [22].

- **Cursor Rollouts** now traces a regression to the offending PR, opens an issue, and lets you start a cloud agent to fix it in one click. Usage credits are included through Oct 3 [23].
- **LangSmith Engine v2** reproduces an issue in the same environment with the same inputs, then builds and tests a fix and iterates on it before handing you a PR [24]. **Managed Deep Agents** now supports memory scoped to a single user or a whole team [25].
- **Codex** has a new `/experimental` flag that keeps the machine from sleeping [26].

Sources

1. software doesn't need to be readable anymore. it needs to be explainable.
2. X post by @GeoffreyHuntley
3. X post by @GeoffreyHuntley
4. X post by @_dhamidi
5. X post by @theo
6. X post by @theo
7. X post by @theo
8. X post by @theo
9. X article by @dkundel
10. X post by @charliermarsh
11. X post by @embirico
12. A total disaster
13. X post by @bridgemindai
14. X post by @theo
15. X post by @theo
16. Inside-Out AI: Rebuilding Airbnb Behind the Scenes and Across the Guest Experience
17. X article by @jasonzhou1993
18. X post by @jasonzhou1993
19. X post by @ThePrimeagen
20. X post by @codegirl007
21. [AINews] Pi 1.0, Pi Durable, and AIE NYC
22. ChatGPT Dots Is Insane & The Truth About Google's NEW Argon
23. X post by @cursor_ai
24. X post by @LangChain
25. X post by @LangChain
26. X post by @GeoffreyHuntley