

Theo's Haiku 5.5 playbook: Opus orchestrates, Haiku does the legwork, Sol audits

Coding Agents Alpha Tracker

2026-10-09

Theo's Haiku 5.5 playbook: Opus orchestrates, Haiku does the legwork, Sol audits

By Coding Agents Alpha Tracker • October 9, 2026

Theo tested Haiku 5.5 in real agent work and shows where it fits: as a cheap subagent under a stronger orchestrator, not as the model that writes code. Also covered: OpenAI's GPT-6.1 Sol Ultrafast with instant steering, three new open-source agent sandboxes, and practitioner notes on reviewing at the module level and running overnight agent PR loops.

Haiku 5.5 in practice: let the smart model delegate to it

Theo's Haiku 5.5 video picks up from yesterday's pricing story and asks where the model actually fits. His answer is that it should not write your code. "Coding is not what makes LLMs expensive." The cost sits in gathering context, verifying, testing and regenerating, and Haiku helps by doing those parts cheaply [1]. It pays off on tasks "where more attempts is more valuable than smart attempts," as long as each result is cheap to check [1]. Anthropic's egg-drop demo makes the case. Opus 5.5 alone took 3.5 minutes, 25 attempts and \$0.47. Opus directing 10 Haiku subagents took under a minute, 86 attempts and \$0.14 [1].

He also shows how it goes wrong when Haiku runs alone. Auditing T3 Code's roughly 1,500 open PRs, Haiku used up most of his GitHub quota. Its script saved the error body as if it were data, and the thread stopped without scheduling its own retry [1]. Two changes fixed the run: - Prompt: "use lots of sub-agents and workflows... use Haiku 5.5 for all sub-agents... minimize work that you do yourself outside of actually getting the data and orchestrating the sub-agents." - Run that prompt on Opus. Opus split the work into batches of about 15 PRs per Haiku agent, roughly 100 agents in all [1].

He plans to edit his CLAUDE.md so it stops forcing every subagent to be Opus or Sol [1].

His default setup stays the same. Skills have Opus write the code and consult GPT-6.1 Sol before opening a PR. He says Sol “catches all the dumb things Opus might miss” [1].

Cost control: Haiku’s price rises 5x above 100k tokens [1]. Theo relays a tip from Anthropic’s Lydia: set Haiku 5.5’s auto-compact window to 100k in Claude Code. It is saved per model, so it caps Haiku subagents without touching your other models [1]. Without a cap, his game demo sent 47 of 51 requests over 100k [1].

OpenAI: GPT-6.1 Sol Ultrafast, instant steering, Codex Cloud re-shipped

According to OpenAI, Ultrafast for GPT-6.1 Sol is rolling out in the API, Codex and ChatGPT Work. OpenAI describes it as “near-Astra intelligence at up to 8x faster speeds than Sol Standard” [2]. Tibo says steering is now instant: the model reacts right away to your adjustments, so you can correct course in real time before it wastes effort. He says instant steering and Ultrafast “work very well together” [3]. Yesterday’s brief covered Theo’s approach of steering a fast model as it works instead of batching corrections. This release supports that approach at a lower price tier than Astra.

Tibo also says OpenAI “silently re-shipped codex cloud. It’s pretty good now” [4]. Codex Cloud can now securely reach resources on your Tailscale tailnet [5], so a cloud agent can use your private services.

Three new open-source sandboxes for agents

- **Microsoft MxC** (repo) is a sandboxing library for Windows, macOS and Linux. It uses processcontainer, bubblewrap and seatbelt underneath. Simon Willison calls it “very promising” [6].
- **Microsoft Quicksand** (repo) is a Python library that bundles QEMU, so it can run something like an Alpine Linux container. Willison has tried it on all three operating systems [7].
- **AWS Strands Box** is an open-source sandbox for developers building agents, announced by Marc Brooker [8].

Practitioner workflows

swyx: review modules, not lines. He lets “slop” through inside modules whose overall behavior he fully understands. The danger is having too many black boxes. In his Slack clone, two agents working at different times each wrote their own message-loading path, and the result was an intermittent race condition [9]. He expects engineers to juggle 5–10 parallel efforts and to work from logs, traces, schemas and captured inputs/outputs, turning them into evals [9]. He has two or three employees under performance review for delivering “cloud slop” [9].

Positron: closed-loop chip verification. Thomas Summers gave agents access to internal infrastructure and Cadence Palladium emulators. The agents read the PDFs, wrote their own Markdown cheat sheets, and built their own test harnesses [9]. The loop now writes test programs, finds failures and writes reports that agents and humans review. He says this became possible with GPT-6 Astra [9]. Token spend briefly passed salaries, peaking above \$100k/day. Opus 5.5 beat Astra on many of their tests, though not all, at a quarter of the price. He still won't run real development on "anything that is less than the best model" [9].

DHH: daily agent PR cycle. For the Campfire rewrite, he lets agents run for a day, reviews and verifies the submitted PRs, merges what is ready, then asks for architectural improvements to each implementation [10]. Benchmarking and checking live in a separate verification project, and the latest performance report is public (report) [11].

Riley Brown: one skill, one-shot video edit. He spent about an hour writing a short-form editing skill, then told Opus 5.5 to "use the skill and edit this video." It removed the background, transcribed the video, added subtitles, built the animations and sound effects with custom JavaScript on an HTML canvas, and checked the result by having Gemini watch it [12]. To develop the skill, he asks for "10 ways you would fill in the animations" for each segment and picks his favorites [13]. Kent C. Dodds separately calls Opus 5.5 "so good at building good looking UI and interactions" [14].

Snyk: fix bad traces from the IDE. Snyk's coding agents use the LangSmith MCP server to investigate a bad production trace and turn it into a dataset without leaving the IDE [15]. Every PR runs the real agent against eval suites and is blocked if it misses thresholds committed to the repo [15].

Smaller items

- **Anthropic OSS Scanner:** Anthropic will periodically scan opted-in open-source projects for vulnerabilities at no cost. Reports include a proof of concept, an explanation and a suggested fix [16].
- **T3 Code** nightly supports many harnesses through ACP, plus a beta Muse provider [17].
- **Kody v2026.10.08:** packages can push events to MCP clients once a topic is opted in with `"mcp": true` [18]. Kent says you can wake ChatGPT through MCP Events, and Codex can pull skills stored in Kody through the MCP Skills extension [19].
- **Cursor /visualize** builds charts inline in the Agents Window [20]. Follow-up questions in the same chat produce new charts [21].
- **ttok 1.0** now defaults to the GPT-5/GPT-6 tokenizer. OpenAI hasn't confirmed that GPT-6 uses the same one, but one experiment found identical counts across seven models [22].
- **Kent C. Dodds** argues that "the best way" to work with agents goes

stale within months because agents get trained on today's best practices. He is focusing on skills that take agents longer to learn [23, 24].

Sources

1. finally a good small model
2. X post by @OpenAIDevs
3. X post by @thsottiaux
4. X post by @thsottiaux
5. X post by @Tailscale
6. X post by @simonw
7. X post by @simonw
8. X post by @MarcJBrooker
9. AI:AM: A Level We Shouldn't Pass? Notes from The Curve + Tokens vs. Salaries & Is SaaS Cooked?
10. X post by @dhh
11. X post by @dhh
12. X post by @rileybrown
13. X post by @rileybrown
14. X post by @kentcdodds
15. How Snyk Turned an Internal Support Agent into a Customer Feature
16. X post by @AnthropicAI
17. X post by @theo
18. X post by @kodykoala
19. X post by @kentcdodds
20. X post by @cursor_ai
21. X post by @cursor_ai
22. ttok 1.0
23. X post by @kentcdodds
24. X post by @kentcdodds