Multi-Agent Control Planes, Local DS4, and Practical Debug Loops

Coding Agents Alpha Tracker

2026-05-12

Multi-Agent Control Planes, Local DS4, and Practical Debug Loops

By Coding Agents Alpha Tracker • May 12, 2026

Today's brief is about orchestration getting real: multi-agent control planes, parallel debugging, model routing, and measurable migrations are beating vague one-agent heroics. Also worth your attention: Cursor's latest product moves, fresh benchmark data on model/harness combos, and DS4's early local-agent momentum.

TOP SIGNAL

- Today's clearest practical shift: multi-agent control is becoming operational. Anthropic shipped Claude Code's `agent view` as a research preview; catwu shared the control-plane flow — run `claude agents`, then hit `<-` in any CLI session, ideally from the repo root — and Boris Cherny called it the best way to level up from one agent to many [1, 2, 3]. Theo's production outage story shows why that matters: he pasted the first error into one agent, opened another for a narrower DB-integrity question, then used the agent to draft schema-aware cleanup SQL while keeping the actual system understanding and final judgment on the human side [4].

TRY THIS

- **Run parallel theory agents on incidents (Theo).** 1) Paste the first prod error into one agent. 2) Start a second agent/tab with a narrower diagnostic question — Theo used `Which of these table referential integrity checks is the most likely to have a problem?` 3) While both run, inspect logs/code yourself. 4) Once you isolate the cause, give the agent your real schema and ask it to draft the exact cleanup SQL, then review before executing [4].

- **Spike before you spec; pseudocode before you generate (Theo).** Build the minimum viable shape first, learn what is annoying or underspecified, then write the real spec from what the spike teaches you. Theo also likes having the model draft pseudocode first, editing it in a back-and-forth, then asking what it looks like in the codebase [4].
- **Route between models/providers instead of marrying one agent (Theo).** Keep more than one subscription/provider live; Theo says he regularly has GPT-5.5 debug Opus output and Opus improve GPT-5.5 UI work, and likes tools that let him hop between Claude/Codex/Cursor/OpenCode when reliability or fit changes [4]. His cost note is the kicker: if prior-tier intelligence is enough, GPT-5.5 Medium matched earlier highs at under half the prior cost [4].
- **Aim agents at boring, benchmarkable migrations.** Igor Alexandrov used Claude to rewrite SafariPortal's tests from RSpec to Minitest and cut local runtime from 16m52s for 7003 examples to 111s for 5698 runs / 19375 assertions; DHH's point is that conversion work makes the upside obvious. Pick one slow suite or framework seam, migrate it, and measure before/after instead of asking for greenfield magic [5, 6].

WHAT SHIPPED

- **Claude Code — `agent view` (research preview).** One list of all sessions, live now; operator flow is `claude agents` + hit `<-` in any CLI session to register it, preferably from the repo root so every agent sits under one control plane [1, 2]. Announcement [2]
- **Cursor Bugbot — effort levels.** Usage-based Bugbot now exposes configurable thinking depth; Cursor says default-effort issues are resolved at merge time more than 80% of the time, and high effort finds 35% more bugs at the same resolution rate. Cursor uses high effort on infra/backend changes and default elsewhere [7, 8, 9]. Docs [7]
- **Cursor for Microsoft Teams.** Mention `@Cursor` in a channel to delegate a task or pull info into Teams; Cursor says it reads the whole thread before implementing and opening a PR for review [10, 11]. Changelog [11]
- **Artificial Analysis Coding Agent Index.** New benchmark mix covers SWE-Bench-Pro-Hard-AA, Terminal-Bench v2, and SWE-Atlas-QnA [12]. Top scores: Opus 4.7 / Cursor CLI 61; GPT-5.5 / Codex 60; Opus 4.7 / Claude Code 60; GPT-5.5 / Cursor CLI 58 [12]. The operational spread matters more than the leaderboard: cost/task varies by more than 30x ($0.07 Composer 2 / Cursor CLI vs. $2.21 GPT-5.5 / Codex), time/task by more than 7x (~6 min Opus 4.7 / Claude Code vs. ~40 min Kimi K2.6 / Claude Code), and the best open-weight result here is GLM-5.1 / Claude Code at 53 [12].
- **Dwarfstar 4 (DS4).** Salvatore Sanfilippo's local DeepSeek v4 stack is

explicitly shaped around coding agents: model-specific inference kernels, a server tailored to agent workflows, disk-backed KV cache with checkpointing, directional steering, and repeated correctness checks against online logits / higher quants [13]. Early signal is solid: Salvatore says he uses it daily with PyAgent/OpenCode, and Armin Ronacher says recent fixes let it build and iterate on a small TUI Tetris game and explain `ds4.c` decisions well enough to feel useful [13, 14, 15].

- **PI + Warden.** Armin says Arendelle acquired Mario's open-source PI to steward it responsibly while keeping it useful as a building block for other agents; the design target is the earlier, more minimal Claude Code behavior that adapts per project [16]. In the same orbit, Sentry's Warden uses Claude Code SDK v1 plus skills to loop on vuln discovery and reportedly found ~100 issues in Sentry [16].
- **Codex computer-use is creeping into setup work.** Peter Steinberger says Codex noticed a missing Google Cloud API while he was adding features to gogcli.sh and started Computer Use to click around Google Cloud Admin to enable it [17].

GO DEEPER

- **Theo on parallel outage debugging (25:37-26:07).** One of the best short demos of using a second agent for a narrower theory instead of waiting for the first one to finish. Good template for DB and incident work where you still hold the map of the system [4].

We all fell for it… (25:36)

- **Theo on the minimum-viable-shape method (41:51-42:25).** If you only watch one planning clip today, make it this: use a quick spike to discover the real constraints, then write the spec after the learning happens [4].

We all fell for it… (41:50)

- **Salvatore on why DS4 got traction (21:32-24:37).** The useful bit is the product framing: faster local inference is not enough; the stack has to behave like a usable coding-agent system end to end [13].

Lo strano clamore suscitato da DS4 (Dwarf Star 4 :) (21:31)

- **Armin/Ben on Warden and agentic security scanning (19:27-21:20).** Good short segment if you care about where harnesses go after coding assistants: Claude Code SDK loops, custom skills, and a focused vuln-finding workflow [16].

State of Agentic Coding #6 with Armin and Ben (19:27)

- **Study the artifacts, not just the hot takes.** Armin shared the full DS4 Tetris trace here: session log [18]. For tiny agentic scripts, Simon Willison's shebang TIL is worth copying line-for-line: TIL [19].

Editorial take: the edge is shifting from bigger prompts to better orchestration — control planes, parallel hypotheses, model routing, and hard before/after checks on performance and maintenance [1, 4, 5, 20].

Sources

1. X post by @claudeai
2. X post by @_catwu
3. X post by @bcherny
4. We all fell for it…
5. X post by @igor_alexandrov
6. X post by @dhh
7. X post by @cursor_ai
8. X post by @cursor_ai
9. X post by @cursor_ai
10. X post by @cursor_ai
11. X post by @cursor_ai
12. X post by @ArtificialAnlys

13. Lo strano clamore suscitato da DS4 (Dwarf Star 4 :)
14. X post by @mitsuhiko
15. X post by @mitsuhiko
16. State of Agentic Coding #6 with Armin and Ben
17. X post by @steipete
18. X post by @mitsuhiko
19. Using LLM in the shebang line of a script
20. Quoting James Shore