Ralph Loops Go Mainstream, Security Agents Ship, and Codex Speeds Up

Coding Agents Alpha Tracker

2026-05-01

Ralph Loops Go Mainstream, Security Agents Ship, and Codex Speeds Up

By Coding Agents Alpha Tracker • May 1, 2026

Goal-persistent agent loops are turning into mainstream product features, while security agents, faster Codex workflows, and sharper harness tactics are giving developers more practical leverage. Today's brief focuses on the patterns practitioners can actually copy: external state, model routing, long-running loops, and small configuration changes that materially improve results.

TOP SIGNAL

The strongest signal today: **goal-persistent agent loops are moving from practitioner hack to product feature**. OpenAI shipped `/goal` in Codex CLI 0.128.0—its take on the Ralph loop—while Addy Osmani's long-running agents writeup lays out the durable recipe underneath: external state, explicit done-conditions, separate evaluator roles, and append-only logs for recovery [1, 2]. Cursor engineer Jediah Katz makes the same point from the harness side: orchestration, context, routing, transport, state, and execution all matter, and a weak layer can tank agent quality [3].

TOOLS & MODELS

- **Codex CLI 0.128.0** — `/goal` keeps a goal alive across turns until completion or token-budget exhaustion. Embiricos says it has shipped to CLI and is coming to the app for all users; Simon Willison notes the behavior is largely driven by the `goals/continuation.md` and `goals/budget_limit.md` prompts [1, 4, 5].
- **Codex app update** — Dynamic task-specific UI, browser/artifact/code annotation, and faster computer use are the main upgrades. OpenAI team posts cite **20% faster** computer/browser use, a **42% faster** Computer

Use benchmark on one workflow, a new device toolbar for responsive testing, and additional browser speed plus Windows fixes [6, 7, 8, 9].

- **OpenClaw v2026.4.29** — Better group chats, follow-up commitments from context, safer exec/pairing/owner controls, NVIDIA provider + model catalogs, faster startup, and plugin/channel fixes. Peter Steinberger says the new group chat finally feels agent-native [10].
- **Security agents are becoming default product surface**:
  - **Claude Security public beta** — Built into Claude Code on the web; point it at a repo, get validated vulnerability findings, and fix them in the same place [11].
  - **Cursor Security Review** — Adds always-on `Security Reviewer` for PRs and `Vulnerability Scanner` for scheduled codebase scans, with configurable triggers, instructions, tooling, and output sharing [12, 13, 14].
- **Model/tool comparison from active use** — Theo says **GPT-5.5** is faster and more likely to unblock him, but can get stuck and choke on context; **Opus 4.7** has better intent/taste but sometimes takes bizarre paths and ignores obvious answers. He also says **Codex feels much faster than Claude Code** on TTFT, TPS, token usage, and tool efficiency [15, 16].
- **LangChain DeepAgents deploy** — Simple cloud deployment for an agent harness via `deepagents.toml`, split into `agent`, `sandbox`, `auth`, and `frontend` sections [17].

WORKFLOWS & TRICKS

- **The long-running loop recipe, stripped to essentials**:
  1. Write a task file with explicit completion criteria (`prd.json` / feature list) before the run starts [2].
  2. For each cycle: pick the next task, build the prompt with relevant context and persistent notes, call the agent, run tests/checks, append to `progress.txt`, update task status, repeat [2].
  3. Keep state outside the model; use append-only logs for recovery/debugging, and split planner/worker/judge or generator/evaluator roles so the model is not grading its own homework [2].
  4. For overnight jobs, run in a worktree, surface lint/typecheck failures back to the agent, and commit progress at meaningful milestones [2].
- **Budget control is now harness design** — Teams in production are seeing token spend rise fast, so the practical playbook is: use cheaper defaults for simple tasks, cap or pool spend for expensive models, and measure spend vs. outcomes monthly. One team cut cost **30%** by changing default model routing; another is actively blocking/managing the most expensive Cursor models and moving to pooled spend [18]. Counterpoint: at least one team refuses anything below **Opus 4.7** for coding because cheaper errors in prod can cost more than the token bill [18].

- **OpenClaw tuning that sounds small but matters** — If group chats felt messy before, retry with **visible replies** enabled and switch from GPT to the **codex harness** plugin. @steipete says that combo materially improved results [19].
- **If you build developer tools, design for the agent as the user** — Patrick Collison says agents are even hungrier for good DX than developers, and Romain Huet puts it more bluntly: the primary developer on your API is an agent like Codex [20, 21]. Stripe's concrete demo bar is high: Claude Code was pointed at `https://github.com/stripe/link-cli` and used secure single-use tokens to make a purchase on Gumroad [20].

PEOPLE TO WATCH

- **Addy Osmani** — Strongest practical synthesis in today's notes on long-running coding agents: external files, explicit done-conditions, and append-only logs [2].
- **Jediah Katz** — Cursor builder with a useful corrective: first-party lab harnesses do not automatically win, and a good agent stack has at least six layers to tune [3].
- **Theo** — High-signal for current model ergonomics because he compares failure modes, not just wins [15, 16].
- **Andrej Karpathy** — Worth tracking for the framing shift from vibe coding to agentic engineering, plus his emphasis on LLM-legible systems and the skill set around them [22, 23].
- **swyx** — Useful operator signal that a tiny team can lean hard on agents in real operations: he says `ai.engineer` serves ~1m unique developers monthly, and his stack includes OpenClaw personally plus Devin and TownAI on the team side [24].

WATCH & LISTEN

- **1:11-1:48 — Starter template to working app.** Fast demo of a useful loop: click `I'm feeling lucky`, let the model plan the logic, then shape the result with multi-turn prompts [25].

Vibe Coding AI Apps (1:11)

- **5:17-5:50 — Self-correcting loop + inline code feedback.** Voice ideas become code, the system fixes its own runtime bugs, and the live API suggests more semantic HTML [25].

Vibe Coding AI Apps (5:17)

- **Full talk to queue** — AIE EU closing note: swyx on using agents to run `ai.engineer` as a tiny team serving ~1m monthly developers [24].

PROJECTS & REPOS

- **snarktank/ralph** — Still the clearest inspectable reference for the long-running loop: task list, prompt build, agent call, tests, progress log, repeat. The important signal today is that major products are scaling this pattern rather than replacing it [2, 1].
- **snarktank/compound-product** — Extends Ralph into chained analysis/planning/execution loops; a good repo to study if you want multiple agent roles without burying the orchestration [2].
- **Codex's `/goal` prompt files** — goals/continuation.md and goals/budget_limit.md are worth reading because they show goal persistence implemented through inspectable prompt files [5].
- **OpenClaw v2026.4.29** — Fast-moving open-source agent surface with a meaningful release this week: better group chat, follow-up commitments, safer exec controls, NVIDIA provider support, and startup/plugin fixes [10].

Editorial take: the durable edge is moving above the model—persistent goals, external state, verification, routing, and recovery are what separate agents that demo well from agents that actually finish the job. [2, 3]

Sources

1. X post by @fcoury
2. Long-running Agents
3. X post by @jediahkatz
4. X post by @embirico
5. Codex CLI 0.128.0 adds /goal
6. X post by @thsottiaux
7. X post by @ajambrosino
8. X post by @AriX
9. X post by @JamesZmSun
10. X post by @openclaw
11. X post by @_catwu
12. X post by @cursor_ai
13. X post by @cursor_ai
14. X post by @cursor_ai
15. X post by @theo
16. X post by @theo
17. X post by @hwchase17
18. The Pulse: token spend breaks budgets – what next?
19. X post by @steipete
20. X post by @patrickc
21. X post by @romainhuet
22. X post by @stephzhan
23. X post by @karpathy
24. X post by @swyx
25. Vibe Coding AI Apps