Spec-First Agent Loops Win as Codex Adds Steering and Subagents

Coding Agents Alpha Tracker

2026-03-29

Spec-First Agent Loops Win as Codex Adds Steering and Subagents

By Coding Agents Alpha Tracker • March 29, 2026

Today's sharpest pattern is constrained autonomy: spec-first loops, separate evaluator models, and tracing dashboards are outperforming blind one-shot runs. Also inside: Codex's latest orchestration features, the secure MCP/OAuth pattern, and the repos/operators worth copying.

TOP SIGNAL

The best builders are shifting from one-shot agent runs to **constrained autonomy**. At the RALPH hackathon, the strongest pattern was spec-first loops plus hard controls—feature-by-feature execution, capped retries, separate evaluator models, and dashboards showing every commit, score, token burn, and failure. Matt Webb's counterpoint is the right guardrail: agents can brute-force a problem with persistent loops, but maintainability still comes from architecture, libraries, and clean interfaces [1, 2].

> "The thing about agentic coding is that agents grind problems into dust." [2]

Translation: let the agent grind, but only inside a harness you can inspect [1, 2].

TOOLS & MODELS

- **Codex app is turning into an orchestration surface.** Romain Huet demoed **GPT-5.4** and **GPT-5.3 Codex Spark** inside the app, with Spark handling an app-translation task in seconds. The bigger change is the control layer around the models: **plugins**, **steering**, **subagents**, and **automations** [1].

- **Practical Codex integrations are getting real.** Huet showed plugins pulling context from Slack, Notion, Gmail, and Figma; he also showed a zero-shot Figma-to-code flow and described scheduling recurring jobs like PR review or comment fixes [1].
- **Remote execution is the missing piece—and it's coming.** Huet said remote connections/dev boxes are being built so Codex can keep running after you close the laptop [1].
- **Codex review is becoming a second-pass checker across tools.** OpenAI uses Codex review on every internal PR, Huet said it often catches things engineers miss, and Codex can also fix issues directly in the PR. Separately, David Marcus said running Codex review from Claude Code finds critical issues "100%" of the time in his workflow, and Huet replied: "This happens a lot!" [1, 3, 4]
- **Model-side pattern to watch: train inside the real harness.** Phil Schmid's summary of Kimi, Cursor, and Chroma reports points to a shared recipe for vertical agentic models: start with a strong base model, train in the production harness, and optimize on outcome-based rewards. The learned behaviors are exactly the ones practitioners want in prod—parallel subagents, self-summarization, and context pruning mid-search [5].
- **If you want prompts instead of blank-page startup friction:** Romain launched a Codex use-cases gallery where starter prompts open directly in the app, including an iOS flow with SwiftUI skills packaged as a plugin. Use cases [6, 7]

WORKFLOWS & TRICKS

- **Spec-first RALPH loop.** 1) Write the spec up front. 2) Break work into a sequence of features. 3) Let the loop walk the spec one item at a time. 4) Treat manual edits as an exception. The hackathon format literally penalized touching the code, which is a useful forcing function if you want better prompts instead of endless agent babysitting [1].
- **The strongest autonomy pattern today: executor + evaluator + retry cap.** AgentForge's recipe was: list 30 features, let **Codex 5.3** code each feature, grade each commit with **Sonnet** on three metrics, cap each feature at 3 iterations, auto-deploy to Vercel, and inspect a dashboard with per-feature time/token/cost data. Result: the dashboard itself was built in **78 minutes**, across **30 commits**, for **$0.95** total [1].
- **Cross-model review loop.** Generate or refactor code in your main tool, then run a **Codex code review** before merge. Huet said OpenAI uses Codex review on every PR and Codex can auto-fix issues it finds in-place; the Marcus/Huet exchange suggests this works especially well as a second-pass review from another environment like Claude Code [1, 3, 4].
- **Secure MCP/API integration pattern, clarified.** Kent C. Dodds said this was **Claude Desktop**, not Claude Code: he prompted it with `Use Kody to create an app that can control my Spotify`, and the generated MCP app handled OAuth without exposing tokens or client

secrets to the model. Kent said the same pattern would work in any MCP-supporting client. Demo: Kent's segment [8, 9, 10, 11]

- **Don't confuse more loops with better software.** Matt Webb's practical correction: agents are good at grinding through a problem with persistent loops, but maintainability and composability still come from architecture—good libraries, clean interfaces, and humans paying attention to system shape while the agent does the grunt work [2].

PEOPLE TO WATCH

- **Romain Huet** — best current window into where Codex is actually going: plugins, steering, subagents, automations, GitHub code review, and remote execution next [1].
- **Kent C. Dodds** — high-signal on safe agent/API workflows because he showed the secret boundary, not just the happy-path demo [8, 9].
- **Phil Schmid** — worth watching if you care about how agentic behavior is getting trained, not just prompted [5, 12].
- **Matt Webb** — useful antidote when the discourse gets too token-maximalist; his architecture-first framing is the right check on brute-force agent loops [2].
- **Addy Osmani** — concise push to aim bigger: if agents are only making your old workflow faster, the side-project may be too small. He pointed to _chenglou's pure TypeScript text-measurement work as the kind of ambitious build that fits the moment [13, 14].

WATCH & LISTEN

- **4:23-5:43 — RALPH loops, fast.** One-minute explanation of the format: specs first, loop second, and a 10-minute penalty if you touch the code. Good mental model if you're designing your own hands-off build loop [1].

The Hackathon Where Humans Can't Code (4:23)

- **11:28-17:37 — AgentForge demo.** The most concrete autonomy demo in the set: 30 features, three-attempt cap, Codex as executor, Sonnet as grader, Vercel deploys, and a per-feature cost/time dashboard built by the agent itself [1].

The Hackathon Where Humans Can't Code (11:27)

PROJECTS & REPOS

- **AgentForge** — public GitHub + live Vercel demo for a RALPH-style harness with commit scoring, per-feature time/token/cost tracking, and hybrid orchestration (RALPH outer loop + auto-research inner loop). Biggest signal: the system built its own dashboard in 78 minutes across 30 commits for $0.95 [1].
- **Ouroboros** — event organizers described the earlier Seoul RALPH-hackathon winner's harness as open source and said it crossed **1k GitHub stars in a week** after launch. Real adoption signal for spec-first autonomous harnesses, not just one-off demos [1].
- **oh-my-open-code / oh-my-claude-code maintainers** — at the event, they said their projects combine for **70k+ GitHub stars** and that agent systems already cover **70-80% of issues and PRs** while they sleep. Worth tracking less for the branding and more for the operating model behind it [1].

Editorial take: the day's best signal is not "full autonomy" by itself; it's ***auditable autonomy****—specs up front, a separate reviewer, architecture doing the heavy lifting, and enough tracing to see where the loop went off the rails* [1, 2].

Sources

1. The Hackathon Where Humans Can't Code
2. Quoting Matt Webb
3. X post by @davidmarcus
4. X post by @romainhuet
5. X post by @_philschmid
6. X post by @romainhuet
7. X post by @romainhuet
8. X post by @kentcdodds
9. X post by @kentcdodds
10. X post by @kentcdodds
11. X post by @kentcdodds
12. X post by @_philschmid
13. X post by @addyosmani
14. X post by @_chenglou