Orchestration-first agent coding: Codex CLI v0.105, spec-driven loops, and eval infra wars

Coding Agents Alpha Tracker

2026-02-27

Orchestration-first agent coding: Codex CLI v0.105, spec-driven loops, and eval infra wars

By Coding Agents Alpha Tracker • February 27, 2026

Today's theme: orchestration beats raw generation. You'll get concrete spec-first workflows from practitioners, major Codex CLI upgrades (v0.105), new PR autofix automation, and why shared eval infrastructure is suddenly a battleground.

TOP SIGNAL

Orchestration is becoming the core dev skill: Addy Osmani argues the enterprise frontier is **orchestrating a modest set of agents with control/traceability**, not running huge swarms [1]. In practice, that shows up as *spec-first work*: Brendan Long's vibe-coding loop starts by writing a detailed GitHub issue ("90% of the work"), optionally having an agent plan, then having another agent implement [2].

TOOLS & MODELS

- **Codex CLI v0.105 (major QoL upgrade)** [3]
  - New: **syntax highlighting**, dictate prompts by holding **spacebar**, better **multi-agent workflows**, improved **approval controls**, plus other QoL changes [3].
  - Install/upgrade: `$ npm i -g @openai/codex@latest` [3].
  - Practitioner reaction: "diffs are beautiful" and it's "very, very fast now" [4].
- **Codex app (Windows) — first waitlist batch invited** [5]
  - Team says they'll "expand from there" as they iterate through feedback [5].
- **Model preference + benchmarkging signals (Codex 5.3)**

  - Mitchell Hashimoto: **Codex 5.3** felt "much more effective" than **Opus 4.6**; after switching back-and-forth, he hasn't touched Opus for a week [6].
  - Romain Huet: **GPT-5.3-Codex** hit **90% on IBench** at **xhigh reasoning**; says with speed gains, "xhigh doesn't feel like a tradeoff anymore" [7].
  - Related run: "decided to run 5.3 codex on xhigh as well, its 90%… rip IBench, survived 3 months" [8].
- **Cursor — Bugbot Autofix (PR issues → auto-fixes)** [9]
  - Announcement: Bugbot can now automatically fix issues it finds in PRs [9].
  - Details: http://cursor.com/blog/bugbot-autofix [10].
- **Devin AI (real production debugging)**
  - swyx reports Devin investigated a production bug (Vercel org migration + forgotten key), asked for exactly what it needed, and verified the fix [11].
- **FactoryAI Droids — "Missions" + terminal "Mission Control"**
  - "Missions": multi-day autonomous goals where you describe what you want, approve a plan, and come back to finished work [12, 13].
  - Mission Control: a terminal view of which feature is being built, which Droid is on it, tools used, and progress [14].
  - Examples FactoryAI says enterprises are running: modernize a 40-year-old COBOL module; migrate >1k microservices across regions; recalc 10 years of pricing; refactor a monolith handling 20M daily API calls with no downtime [13].
- **OpenClaw — new beta bits** [15]
  - Adds: external secrets management (`openclaw secrets`) [15, 16], CP thread-bound agents [15], WebSocket support for Codex [15], and Codex/Claude Code as first-class subagents via ACP [17].
- **Omarchy 3.4 — agent features shipped**
  - Release highlights include "new agent features (claude by default + tmux swarm!)" and a tailored tmux setup [18, 19].
- **Harbor framework — shared agent eval infra momentum**
  - Laude Institute frames Harbor as shared infrastructure to standardize benchmarks via one interface (repeatable runs, standardized traces, production-grade practice) [20].
  - swyx says his team is prioritizing migrating evals to Harbor and calls it dominant in RL infra/evals for terminal agents [21].

WORKFLOWS & TRICKS

- **Spec-driven agent work (make the spec the artifact)**
  - Brendan Long's repeatable loop for large vibe-coded apps:
    1) Write a GitHub issue [2]
    2) If it's complex, have an agent produce a plan and update the issue [2]

3) Have another agent read the issue and implement it [2]
  - He claims a detailed enough issue is "**90% of the work**" and rewriting it is often what fixes problems [2].
- **Enterprise-grade orchestration guidance (modest fleets, strong controls)**
  - Addy Osmani's concrete advice: spend **30–40%** of task time writing the spec—constraints, success criteria, stack/architecture—and gather context in a resources directory; otherwise you "waste tokens" and LLMs default to "lowest common denominator" patterns [1].
  - For teams: codify best practices in context (e.g., MCP-callable systems or even markdown files) to raise the odds the output is shippable [1].
  - He also flags the real bottleneck: "Not generation, but coordination" [1].
- **Close the loop: isolate the runtime so agents can run it**
  - Kent C. Dodds: "**get your app running in an isolated environment** to close the agent loop" [22].
  - He points to his Epic Stack guiding principles—"Minimize Setup Friction" and "Offline Development"—as a practical way to make this easier [22].
- **Hoard working examples, then recombine (prompt with concrete known-good snippets)**
  - Simon Willison's pattern: keep a personal library of solved examples across blogs/TIL, many repos, and small "HTML tools" pages, because agents can recombine them quickly [23].
  - His OCR tool story: he combined snippets for PDF rendering and OCR into a single HTML page via prompt, iterated a few times, and ended with a tool he still uses [23].
  - Agent tip: when asking Claude Code to reuse an existing tool, he sometimes specifies `curl` explicitly to fetch **raw HTML** instead of a summarizing fetch tool [23].
- **Tests aren't a moat anymore (agents can recreate them fast)**
  - tldraw moved tests to a closed-source repo to prevent "Slop Fork" forks [24].
  - Armin Ronacher's counterpoint: agents can generate language/implementation-agnostic test suites quickly if there's a reference implementation [25].
- **Security footnote from a vibe-coded app**
  - In Simon Willison's Present.app walkthrough, the remote-control web server used GET requests for state changes (e.g., `/next`, `/prev`), which he notes opens up CSRF vulnerabilities—he didn't care for that application [26].

PEOPLE TO WATCH

- **Addy Osmani (Google Cloud AI)** — clearest "enterprise reality check": quality bars, traceability, and spec/context discipline, plus a strong stance that orchestration is the thing to learn [1].
- **Simon Willison** — consistently turns agent usage into transferable patterns (Agentic Engineering Patterns + "hoard examples" + codebase walkthrough prompts) [26, 23].
- **Brendan Long** — practical decomposition: write issues like a system design interview, then let agents execute [2].
- **Nicholas Moy (DeepMind)** — framing: "10x engineer" becomes "10 agent orchestrator," measured by concurrent agents you can run effectively [27].
- **Dylan Patel (Semianalysis)** — adoption signal: Claude Code share of GitHub commits going 2%→4% in a month, with a broader estimate of total AI-written code around ~10% [28].

WATCH & LISTEN

1) Addy Osmani: "Learn orchestration" + the path to agent fleets (23:32–25:54)

Hook: Practical roadmap from single-agent prompting to multi-agent orchestration and coordination patterns—before you burn tokens on experimental swarms [1].



Live with Tim O'Reilly: A Conversation with Google Cloud AI Director Addy Osmani (23:31)

2) SAIL LIVE #6: why SWE-Bench got saturated (and what that says about evals) (29:49–33:48)

Hook: A clear explanation of how SWE-Bench is constructed, why it became the default "agentic coding" benchmark, and why that creates problems once it's widely known and reused [29].



Distillation & How Models Cheat | SAIL LIVE #6 (29:49)

PROJECTS & REPOS

- **Agentic Engineering Patterns (Simon Willison)** — a living guide of coding-agent practices and patterns (agentic engineering vs vibe coding framing) [26]
  - https://simonwillison.net/guides/agentic-engineering-patterns/ [26]
- **Present.app (Simon Willison)** — vibe-coded SwiftUI macOS presentation app where each "slide" is a URL; GitHub repo shared [26]
  - https://github.com/simonw/present [26]
- **OpenClaw releases + docs (beta features shipping)** [15]
  - https://github.com/openclaw/openclaw/releases [15]
  - Secrets docs: https://docs.openclaw.ai/cli/secrets [16]
  - ACP agents docs: https://docs.openclaw.ai/tools/acp-agents [17]
- **Cursor Bugbot Autofix announcement + writeup** [10]
  - http://cursor.com/blog/bugbot-autofix [10]
- **Omarchy 3.4 release** (61 contributors; agent features + tmux work) [18]
  - https://github.com/basecamp/omarchy/releases/tag/v3.4.0 [18]
- **tldraw tests move discussion** (tests closed-source) [24]
  - https://github.com/tldraw/tldraw/issues/8082 [24]

Editorial take: The leverage is shifting from "pick the best model" to "build

the tightest loop": spec → isolated runtime → tests/evals → approvals—and only then scale agents.

Sources

1. Live with Tim O'Reilly: A Conversation with Google Cloud AI Director Addy Osmani
2. Vibe Coding is a System Design Interview
3. X post by @thsottiaux
4. X post by @iannuttall
5. X post by @thsottiaux
6. X post by @mitchellh
7. X post by @romainhuet
8. X post by @adonis_singh
9. X post by @cursor_ai
10. X post by @cursor_ai
11. X post by @swyx
12. X post by @FactoryAI
13. X post by @matanSF
14. X post by @FactoryAI
15. X post by @steipete
16. X post by @steipete
17. X post by @steipete
18. X post by @dhh
19. X post by @dhh
20. X post by @LaudeInstitute
21. X post by @swyx
22. X post by @kentcdodds
23. Hoard things you know how to do
24. X post by @cramforce
25. X post by @mitsuhiko
26. Agentic Engineering Patterns
27. X post by @thenickmoy
28. Dylan Patel Explains the AI War While Cooking | In-Context Cooking
29. Distillation & How Models Cheat | SAIL LIVE #6