Spec Loops and Small-Task Discipline Reset the Coding-Agent Playbook

Coding Agents Alpha Tracker

2026-03-16

Spec Loops and Small-Task Discipline Reset the Coding-Agent Playbook

By Coding Agents Alpha Tracker • March 16, 2026

Simon Willison's new framing of agentic engineering was the key signal today, and the best supporting evidence came from practitioners showing what disciplined loops look like in practice: Geoffrey Huntley's spec-first porting workflow, Armin Ronacher's small-task model comparison, and ThePrimeTime's warning about agent-driven work sprawl. Also included: CodexBar 0.18, Omarchy's npm wrapper move, and three clips worth watching.

TOP SIGNAL

Simon Willison's new "What is agentic engineering?" chapter is the clearest practical reset today: coding agents matter when they can *write and execute* code in a tool loop toward a goal, not when they just autocomplete text. The actionable part is his operating model—give the agent the right tools, describe the task at the right level, verify the result, then update instructions and the harness as you learn because the model will not learn from yesterday's mistakes on its own [1]. Geoffrey Huntley's citation-driven porting loop and ThePrimeTime's side-project experience point the same way: harness design beats raw output [2, 3].

> "LLMs don't learn from their past mistakes, but coding agents can, provided we deliberately update our instructions and tool harnesses to account for what we learn along the way." [1]

TOOLS & MODELS

- **CodexBar 0.18** — new providers (**Kilo**, **Ollama**, **OpenRouter**), Codex historical pace + risk forecasting + backfill, a merged-menu Overview tab,

fewer Claude keychain prompt annoyances, and lower CPU/energy use with faster JSONL scanning [4]. Release notes [4]

- **Opus vs Codex on small diffs** — Armin Ronacher says that once changes are sufficiently small, there is little to no difference in how **Opus** and **Codex** behave. Good reminder that task decomposition can matter more than model tribalism [5].
- **OpenClaw direction** — Peter Steinberger says the plugin system is being pushed toward a leaner core plus more powerful plugins, with support for **Claude Code/Codex plugin bundles** planned [6].
- **Omarchy's packaging move** — DHH is moving AI tooling out of regular repos and onto **npm** behind an always-updated **npx** wrapper because **opencode** is shipping about **7 releases per day** [7].

WORKFLOWS & TRICKS

- **Spec-first porting loop**
  1. Compress `tests/*` into `/specs/*.md` with separate subagents, linking implementation as citations [2].
  2. Do the same for `src/*`, again linking implementation into the specs [2].
  3. Run another Ralph loop to create a TODO, then execute classic Ralph doing **one thing and the most important thing per loop** [2].
  4. Configure the target language for **strict compilation** [2].
  5. Keep citations in the specs so the agent can study the original implementation during execution while stages 1-2 stay decoupled from the source language [2].
- **Use task size as a quality lever** — Armin's way to fight "slop creep": make the change smaller. His takeaway was that for sufficiently small edits, Opus and Codex behaved nearly the same [5].
- **Treat harness updates as part of the job** — Simon's durable checklist: give agents the tools they need, specify the problem at the right level of detail, verify the result, and then change instructions/tooling based on what failed [1].
- **Don't let cheap MVPs multiply bad work** — ThePrimeTime's warning is operational: faster prompting makes it easy to spin up multiple rough ideas, but each one creates more waiting, babysitting, and cleanup. More code output did **not** mean better code or better problem selection [3].
- **Repo-triage heuristic** — if someone says they "solved" a problem but the GitHub history is only about **48 hours** old, Armin says assume it has not been properly evaluated yet [8].
- **Packaging trick for fast-moving agent deps** — if tool churn is too high to vendor comfortably, split AI tooling out of the main repo and lazy-load the latest version via npm/npx [7].

PEOPLE TO WATCH

- **Simon Willison** — published a foundational chapter defining agentic engineering as software development with agents that write and execute code, and says the guide will keep evolving as patterns mature [1].
- **Geoffrey Huntley** — shared a concrete, citation-driven language-porting workflow instead of a vague "just use agents" take [2].
- **Armin Ronacher** — high signal today for both operator insight (small-task Opus/Codex parity) and ecosystem skepticism (too many flashy products, too little real evaluation) [5, 9, 10, 11, 8].
- **Peter Steinberger** — actively shipping in the tooling layer: CodexBar 0.18 is out, and OpenClaw plugin bundles for Claude Code/Codex are on deck [4, 6].
- **ThePrimeTime** — worth watching for a blunt firsthand report on where agent speed helps, where it hurts, and how easily the work can sprawl past the point of usefulness [3].

WATCH & LISTEN

- **7:49-8:29 — The "Faustian bargain" of fast MVPs**: Best clip today if you're over-spawning agent jobs. ThePrimeTime explains how easy first drafts turn into longer prompt/wait cycles and constant babysitting once multiple experiments are running [3].

What is wrong with us?! (7:49)

- **9:00-9:32 — Output is not the bottleneck**: The punchline is sharp: generating more code did not mean better code, satisfaction, or the right product. The real bottleneck became choosing the right problem [3].

What is wrong with us?! (9:00)

- **11:30-11:49 — Keep the tool in its place**: Short corrective on work/life balance. One more feature is not worth crowding out actual life [3].

What is wrong with us?! (11:29)

PROJECTS & REPOS

- **`CodexBar v0.18`** — adds provider breadth, Codex pace/risk forecasting, backfill, a new overview surface, and lower resource use [4].
- **`Omarchy` AI-tooling commit** — practical repo-maintenance pattern: keep volatile AI tooling out of the main repo and fetch it on demand. The adoption signal is upstream churn: opencode is releasing about seven times per day [7].
- **OpenClaw plugin ecosystem** — watch this if you care about pluginized agent surfaces: steipete is trying to make the core leaner while expanding what plugins and bundled integrations can do [6].

Editorial take: today's edge is not more agent output; it's tighter loop design—specs with citations, smaller task slices, and explicit verification. [1, 2, 5, 3]

Sources

1. What is agentic engineering?
2. porting software has been trivial for a while now. here's how you do it.
3. What is wrong with us?!
4. X post by @steipete

5. X post by @mitsuhiko
6. X post by @steipete
7. X post by @dhh
8. X post by @mitsuhiko
9. X post by @mitsuhiko
10. X post by @mitsuhiko
11. X post by @mitsuhiko