

Anthropic ports its Messages API to Rust with Claude, with no Rust developers at the start

Coding Agents Alpha Tracker

2026-10-28

Anthropic ports its Messages API to Rust with Claude, with no Rust developers at the start

By Coding Agents Alpha Tracker • October 28, 2026

Anthropic explains how it rewrote a production API with an agent, checked by shadow traffic. LangChain has open-sourced a harness built around async sub-agents and code-enforced gates. Also: building Claude Code mods by prompting, and Artificial Analysis's new index with a merge-graded coding benchmark.

Anthropic: humans design, Claude ports, shadow traffic verifies

On Latent Space, Anthropic's platform team said it has fully rolled out a Rust version of the Messages API, replacing a CPU-hungry Python monolith [1]. When they chose Rust, the team had no Rust developers. Claude wrote the code, and humans designed the distributed-systems architecture [1]. They said the two things that mattered most were the plan and the verification. They built a complete Rust fork of the API, ran shadow traffic through it, and compared its outputs with the Python system's [1]. They ported the whole system and then turned it on gradually, instead of the piece-by-piece migration they said would normally take years [1]. They also kept it a pure port and held back new features [1]. They estimated about 40–50% CPU savings and a 10–20% improvement in time to first token [1].

The steps you can copy: write the plan yourself, have the agent do a scope-frozen port, and check it by comparing outputs from the old and new systems on real traffic.

Their other advice from the same episode: - **Evals:** the most common mistake is having too few. Start simple, build a fairly large suite, and let agents help it grow [1]. Treat engineering like research: define the goal as an eval, then check whether a failure comes from the eval, the environment or the model. "A lot of

times your eval is wrong” [1]. - **Multi-agent:** use one lead and split work that can run in parallel across subagents that share context. The subagents should run asynchronously and not block each other. “Other than that, like let it rip” [1]. - **Sandboxing:** Claude Managed Agents keeps the harness (the “brain”) on a durable server, separate from the sandbox (the “hands”). The agent sees only placeholder credentials. An egress proxy injects the real ones from a separate vault, and network access can be limited to an allowlist of domains or turned off [1].

LangChain open-sources its design agent harness

LangChain’s Content Studio is a Slack agent that makes slides, charts and videos. It runs on Managed Deep Agents, and LangChain has published the harness for others to fork [2]. LangChain reports median model spend per request fell from about \$8 to about \$0.50, and its own code shrank about 80% after moving to Managed Deep Agents [3]. Several of the patterns apply directly to coding agents:

- **Everything is code, in a persistent sandbox.** Each Slack thread gets its own Linux sandbox. Slides are HTML, charts are data specs and videos are animation timelines, so an edit can change one value and rerender [2].
- **Rules that must always hold go in code, not the prompt.** The agent sometimes skipped a review step that was only described in its instructions. The pipeline now blocks assembly until every part passes review: “The order that must always hold lives in code, and the judgment stays in the prompt” [2].
- **Skills are scoped to jobs.** A router skill maps request types to specialist skills. Each of the 48 skills reads like a routing rule, including when *not* to use it [2]. MCP tools sit behind search, and their schemas load only when needed [2].
- **The main agent acts as a “chief of staff.”** Subagents run asynchronously. Each gets a brief covering its role, context, which files it owns and how to validate its work, plus a model and effort level chosen for the task. The coordination tools are `start_task`, `check_tasks`, `cancel_task/resume_task`, `report_progress` and `schedule_wakeup` [2].
- **Memory stays small.** AGENTS.md is limited to a 200-line index, after an earlier version grew to 500K tokens. After each job, a subagent records up to five evidence-backed lessons, and a weekly pass promotes the useful ones into skills [2].
- **Remove workarounds when models improve.** After Opus 5.5, they removed most hard-coded rules and kept only principles. They got more variety and less repetition [2].

Harrison Chase’s Interrupt London talk covers similar ground. LangChain’s open-source OpenWiki builds a coding knowledge base from your repo. It can also read Codex or Claude Code traces to find what the agent struggles with [4]. His eval advice: grade the final environment state or the tool-call sequence, not

just the answer [4]. Run evals that write data against a mock or dev environment, not production [4].

Claude Code mods: build them by describing them

Addy Osmani posted a guide to Claude Code mods. Mods are JS/TS files that react to session events, change Claude Code's behavior or draw custom UI [5]. They are hooks chained like middleware. Each hook can observe an event, rewrite it, or answer it so the default behavior never runs [5]. To make one, describe it to Claude, approve the files and enable hot reloading. Then keep asking for changes without restarting [5]. Mods made this way last only for the session. To keep one, copy it from `~/.claude/dev-mods/<session-id>/` and install it as a plugin [5]. The API may change, so use the generated types in `.claude-plugin/types/` as the reference for your version [5].

Example code for Token Weather (context usage), Blast Radius (confirming risky commands) and Replay Theater (per-edit diffs, including subagent edits) is in anthropics/claude-code-playground [5]. Mods are not sandboxed and run with your permissions [5]. Blast Radius only reads the command text, so `$(...)`, aliases and scripts can get past it. Use permission rules if you need a hard block [5].

Benchmarks: a coding test graded on whether it would be merged

Artificial Analysis Intelligence Index v5 adds FrontierCode 2.0, Cognition's private benchmark. It grades real-world coding tasks on whether a repository maintainer would merge the agent's change [6]. On the overall index, Opus 5.5 (max) leads at 58, followed by GPT-6 Astra at 57 and GPT-6.1 Sol at 55. The best released open-weights models score 32 [6]. Among the top 10, cost per task runs from \$0.96 for GPT-6.1 Terra to \$13.04 for Claude Fable 5.1 [6].

Short notes

- **Cursor design mode** added Annotate. You can edit elements, draw on the page and check different screen sizes as visual prompts for the agent [7]. Open it with `Cmd+Shift+D` in the browser (docs) [8].
- **Kent C. Dodds** says he ships dozens of PRs a day with Cursor and rarely opens the Cursor app. He drives it through Grok Bot [9, 10]. When his Cursor credits ran low, he had the bot move work to Devin's cloud agent, with Kody handling the handoff [11]. He says he would use the Claude and OpenAI cloud agents more if they had APIs [12]. On dropping PRs, he says he would first have to rebuild what PRs give him: AI review, preview deploys, CI and a record of decisions [13].
- **Matt Pocock** trusts agents most on tasks where they have all the data, can run experiments, can verify results on their own, and can hill-climb

toward the goal [14].

- **Steve Yegge** says his Wheelhouse software factory produces 2,000–4,000 commits a week and runs without sandboxing. Its constraints are doctrine and policy, human gates and 14 programmatic denials [15]. He warns these setups are expensive, hard to maintain and risky in enterprises [15].
- **T3 Code:** Theo says a change from Cursor broke the latest nightly. He is building a pipeline to prevent this and recommends the stable channel [16, 17].

Sources

1. How Anthropic ships the #1 LLM in the World — with Katelyn Lesse and Angela Jiang
2. How We Built Our Design Agent
3. X article by @LangChain
4. How to Build Agents That Improve Over Time: Model, Harness, Context | Interrupt London
5. X article by @addyosmani
6. X post by @ArtificialAnlys
7. X post by @cursor_ai
8. X post by @cursor_ai
9. X post by @kentcdodds
10. X post by @kentcdodds
11. X post by @kentcdodds
12. X post by @kentcdodds
13. X post by @kentcdodds
14. X post by @mattpocockuk
15. The Counting House
16. X post by @theo
17. X post by @theo