Slack-Native Agents, PR-Proof Loops, and Fresh OSS Coding Agent Kits

Coding Agents Alpha Tracker

2026-07-03

Slack-Native Agents, PR-Proof Loops, and Fresh OSS Coding Agent Kits

By Coding Agents Alpha Tracker • July 3, 2026

Claude Tag's channel-based, persistent-agent model was the biggest workflow shift today. Also worth stealing: Simon Willison's spec-to-TDD agent loop, Peter Steinberger's PR-proof stack, and a tight set of new shipping updates around Artifacts, trace normalization, and routing.

TOP SIGNAL

Anthropic's Claude Tag is the clearest workflow shift today: instead of opening a coding agent only when you need it, you add it to a Slack channel, set standing instructions once, and let it monitor, answer, fix, and follow up over days or weeks with persistent channel memory [1]. Anthropic says its internal version now writes 65% of product-org PRs and is used across eng, product, data, sales, and marketing; Boris Cherny says that changed his own default from using Claude Code for everything to using Tag for simple fixes, data questions, and more team-visible work [1, 2].

TRY THIS

- **Give the agent a standing job, not just a prompt (Cat Wu + Boris Cherny).** Add Tag to a public channel, then set one-time rules in plain English: `always respond to every data question`, `monitor only X issues`, or `answer then react with` . Let it run long-lived tasks and post back fixes or videos in-thread; keep the work public so the team can steer and learn from the same session. Boris says he started with simple fixes and data questions, then kept moving more work to Tag as he got comfortable; if it gets noisy, you can tell it to jump in less or more and it will remember that too [1].

- **Use spec -> commit -> red/green TDD as the agent contract (Simon Willison).** Start with `uvx --prerelease=allow --with llm-coding-agent llm code`, then prompt: `Write a spec.md for this project - it will depend on the latest "llm" alpha from PyPI and implement a Claude code style coding agent complete with tools for reading and editing files and executing commands.` Follow with: `Commit the spec, then build it using red/green TDD in a series of sensible commits...`. If you do not want full `--yolo`, use an allowlist like `llm code --allow "pytest*" --allow "git diff*"` [3].
- **Make PRs carry proof, not just diffs (Peter Steinberger).** After the agent opens a PR, have it attach or use a sanitized transcript JSON as review context; Peter says longer transcripts materially increase confidence that the agent actually understood the work. Then trigger an auto-review skill on every PR or commit that invokes whatever local CLI you already have installed, but route the feedback back into the original coding session and write the accepted review decisions into the PR description [4].
- **Turn one machine into the fleet brain (Theo).** Keep SSH keys on a central box, give the agent a skill that documents every machine and your default configs, and use Linux worktrees plus Railway CLI/MCP setup so subagents can branch, create services, and spin staging or PR environments without tying up your laptop. Add auto-`tmux` on SSH and use T3 Code over Tailscale or local network when you need remote GUI plus screenshots; Theo's examples had ext4 boxes finishing file/install-heavy work much faster than his Mac and staying calm under subagent load [5].

WHAT SHIPPED

- **Claude Tag** — now in Slack with Claude Fable 5 and launch credits: $25k for Enterprise orgs and $2.5k for Team orgs through Sept. 1. Anthropic says the internal version is already used across eng/product/data/sales/marketing and lands 65% of product PRs; Teams is next. Get started: claude.com/product/tag [6, 7, 2, 1].
- **Claude Code Artifacts** — now on Pro and Max plans. Ask for an artifact and Claude publishes a live, private, self-contained page on claude.ai that keeps updating while the agent works; Boris Cherny called them life changing, and Mike Krieger used one to visualize tricky experiment-gate logic for his team [8, 9, 10].
- **llm-coding-agent 0.1a0** — Simon Willison's new OSS Python library for a Claude Code-style agent, with CLI recipes like `llm code --yolo`, a Python `CodingAgent(...)` API, and built-in read/edit/search/write/execute tools. Read the README and commit sequence [3].
- **LangSmith unified coding-agent traces** — LangChain says it now

normalizes Claude Code, Codex, Cursor, Copilot, Pi, and OpenCode sessions into the same trace tree, metadata, and query model, aimed at restoring visibility when teams mix tools and bills spike. Details: langchain.com/blog/fix-your-coding-agent-bill [11].

- **Sakana Fugu** — router/orchestrator model now available in Codex and OpenCode. It picks the best model per task and can recursively rewrite prompts and verify outputs before deciding what to call; Sakana says it has already been used in auto-research and robot-control demos [12].
- **Codex chief-of-threads pattern** — OpenAI says any Codex conversation can spin up independent threads, and teams are using one main thread to delegate PR reviews or talk prep to child threads. Same backend agent, but the non-dev UI can hide diffs and shell commands by default, which matters if you want agent usage to spread beyond engineers [12].

GO DEEPER

- **1:37-2:28 — Claude Tag mental model.** Good quick intro to the product model: proactive in-channel agent, public collaboration, and memory across sessions [1].

The future of work with @Claude (1:36)

- **5:05-7:04 — Peter Steinberger's auto-review skill.** Best short explanation today of how to get a second model's review without losing the original session's context [4].

"My attention is the bottleneck" | Peter Steinberger, OpenClaw (5:05)

- **40:54-42:21 — LangSmith Engine's next hard problem.** Ben Tannehill explains why finding issues from traces is only half the job; the real unlock is proving proposed fixes against evals before surfacing them [13].

The best AI agents are secretly teams (40:53)

- **Repo to study — llm-coding-agent 0.1a0.** Read the README plus commit sequence together; it is a clean spec-first, TDD-driven example of letting an agent build an agent [3].
- **Repo to study — DSPy prompt harness for Datasette Agent.** Simon's harness evaluates the real SQL-answering tools against live Datasette and surfaced a concrete prompt bug: the schema listing omitted column names, which nudged the model into guessing columns and getting stuck in error-retry loops. Start here: github.com/simonw/research/tree/main/dspy-datasette-agent-prompts#readme [14].

Editorial take: the strongest setups today reduce babysitting by giving agents durable context and better proof loops, not by pretending humans are out of the loop [1, 4].

Sources

1. The future of work with @Claude
2. X post by @_catwu
3. llm-coding-agent 0.1a0
4. "My attention is the bottleneck" | Peter Steinberger, OpenClaw

5. Why I'm moving to Linux (for real)
6. X post by @claudeai
7. X post by @_catwu
8. X post by @ClaudeDevs
9. X post by @bcherny
10. X post by @mikeyk
11. X post by @LangChain
12. ThursdAI - live from AI Engineer WF in SF w/ Swyx, OpenAI, DeepMind, Nvidia, Sakana & EXO labs!
13. The best AI agents are secretly teams
14. Using DSPy to evaluate and improve Datasette Agent's SQL system prompts