Inspectable Multi-Agent Workflows: Manager, CI Run Loops, and Patch.md

Coding Agents Alpha Tracker

2026-04-16

Inspectable Multi-Agent Workflows: Manager, CI Run Loops, and Patch.md

By Coding Agents Alpha Tracker • April 16, 2026

Today's strongest signal was the move toward inspectable orchestration: Manager launched as a Unix-style multi-agent CLI, while practitioners shared tighter loops for CI, portable memory, repo-aware artifact building, and open-source customization. The sharpest model takeaway was equally practical: frontier models still lead for coding agents, and open-model claims get shaky fast on complex security reasoning.

TOP SIGNAL

The real shift is from hidden subagents to *inspectable orchestration.* Imbue's new MIT-licensed `manager` launched today as a CLI for creating, listing, attaching to, viewing transcripts from, and messaging 1,000+ Claude Code agents in parallel on top of tmux/SSH/Git/Docker, with remote runs via Modal and no backing service or database [1]. Harrison Chase describes the same direction as moving from one agent to an agent that manages five agents, and swyx frames it similarly: subagents are mostly an optimization problem, while boss-agents that compose/manage other agents are the real capabilities jump [1, 2]. Josh, Imbue's CTO, says he has been shipping about 10,000 lines of code per day since December using these workflows, and his Claude spend last month looked like roughly 3 FTE while producing more value than that in code [1].

TOOLS & MODELS

- **`manager` (new, MIT):** tiny surface area, big leverage. Core commands are `create`, `list`, `attach`, `transcript`, and `message`; you can run agents remotely with `--provider modal`, and `manager ask` uses an agent to read the docs and explain usage. The design choice matters: it is built on

tmux/SSH/Git/Docker so both humans and agents can inspect and debug it [1].

- **LangChain's stack split is getting clearer:** `Deep Agents` is the open, model-agnostic harness; `LangGraph` is for more directed workflows; `LangSmith Fleet` is the no-code layer for long-running agents with human-in-the-loop interaction. Async subagents are the next step Harrison called out [1].
- **OpenClaw hardened its security model:** after 4 months and thousands of work hours, it now supports yolo mode, Docker or OpenShell sandboxing, allow-lists, and per-access exec allow/deny prompts. Peter Steinberger says hundreds of security researchers have pen-tested it [3].
- **Model pecking order remains pragmatic:** Harrison says OpenAI, Anthropic, and Google still drive agents best today; GLM5 and Minimax 2.7 are the most promising open models, and GLM5 is already used internally for some coding agents [1]. Salvatore Sanfilippo's separate GPT OSS 120B test on a complex OpenBSD bug failed across multiple runs to recover the actual causal chain, which is a useful caution if you are evaluating open models for security auditing [4].

WORKFLOWS & TRICKS

- **Scale agents in layers, not all at once:** 1) start with one Claude Code session, 2) add more terminals, 3) put repetitive work in cron, 4) when local cores become the bottleneck, offload tests to Modal sandboxes, 5) when review becomes the bottleneck, add reviewer agents and stop-hooks, and 6) collapse bug fixes into stacked single-commit changes so humans can review them fast [1].
- **Lock the agent to a failing test before it writes code:** Theo's recipe is to use rwx's run loop so the agent can run your full CI locally with caching before it commits, then pair that with a commit that adds the failing test for the feature you want. His example went from a 2 minute setup run to 22 second cached checks [5].
- **Keep core memory portable and dumb:** Harrison's advice is plain files and open standards like `agents.md` and skills, not overbuilt memory stacks. For core memory, simple append/update strings or markdown files beat rushing to a graph DB or vector DB, and he gives a concrete warning: losing his EA agent's learned memory made it materially worse [1].
- **Use Claude as a repo-aware artifact builder:** Simon Willison's exact prompt was `Clone https://github.com/simonw/datasette.io and look at the news.yaml file and how it is rendered on the homepage. Build an artifact I can paste that YAML into which previews what it will look like, and highlights any markdown errors or YAML errors.` That produced a working side-by-side preview/editor UI for a real project and removed some maintenance friction [6, 7]. Read the writeup here: https://simonwillison.net/2026/Apr/16/datasette-io-preview/ [6].

- **Theo's `Patch.md` idea is worth stealing even before it ships:** keep a text file that records the *intent* of every local customization to a fork. When upstream updates break your changes, let an agent first try to resolve the merge, and if that fails, reapply the intended features from `Patch.md` to the new version [5].

PEOPLE TO WATCH

- **Josh + Harrison Chase:** best practical long-form of the day on what actually breaks as you add more agents: cores, observability, review throughput, memory, and tool choice [1].
- **Theo:** his open-source thesis is backed by real usage, not ideology. T3 Code has about 42k installs, 16k weekly actives, ~9k GitHub stars, and 1.5k forks, with users already adding split chat, queueing, tmux integration, and handoff flows in their own forks [5].
- **Simon Willison:** still one of the best sources for bounded, reproducible Claude workflows that solve real developer pain instead of demo problems [6, 7].
- **Salvatore Sanfilippo:** strong antidote to benchmark theater. His GPT OSS 120B test asks the right question for bug hunting: did the model understand the state transition that causes the bug, or did it just guess plausible bug classes [4]?
- **Armin Ronacher + swyx:** useful framing pair for the week. Armin says tmux is great software for an agent but rough day-to-day UX for humans, while swyx argues the bigger leap is not more hidden subagents but agents that manage other agents [8, 9, 2].

WATCH & LISTEN

- **Manager demo — 04:06-07:54.** Fastest way to grok today's biggest pattern: create local or remote agents, inspect transcripts, and wire agent-to-agent messaging from plain Unix primitives [1].

How to usefully run 1,000 agents in parallel (4:06)

- **Theo on `Patch.md` — 32:51-35:43.** One of the most interesting near-term ideas for AI-customized software: capture the intent of your fork in a text file, then let an agent resolve or reapply those changes when upstream updates break them [5].

I think every company should open source their code. (32:49)

PROJECTS & REPOS

- **T3 Code:** open-source GUI wrapper for Claude/Cursor/Codex CLIs with about 42k installs, 16k weekly active users, ~9k GitHub stars, and 1.5k forks. The fork rate is the signal: users are actively customizing the product instead of just consuming it [5].
- **Manager:** brand-new MIT-licensed CLI for parallel Claude Code agents, designed to be inspectable and scriptable instead of service-backed [1].
- **OpenClaw:** open-source coding agent tool with a much more mature security posture than it had in December, including sandbox choices, allow-lists, and exec prompts [3].

Editorial take: the durable edge right now looks less like agent magic and more like good systems engineering — simple harnesses, failing tests, portable memory, and explicit supervision around lots of small workers. [1, 5]

Sources

1. How to usefully run 1,000 agents in parallel
2. X post by @swyx
3. X post by @steipete

4. No, gpt 120b non sa trovare il bug di OpenBSD
5. I think every company should open source their code.
6. X post by @simonw
7. datasette.io news preview
8. X post by @mitsuhiko
9. X post by @mitsuhiko