# Subagent Routing, Review-as-Verification, and Fable in the Wild

Coding Agents Alpha Tracker

2026-07-04

## Subagent Routing, Review-as-Verification, and Fable in the Wild

*By Coding Agents Alpha Tracker • July 4, 2026*

The strongest coding-agent pattern today is simple: keep the main model on judgment, route implementation to lower-power workers, and tighten verification before merge. This brief packs exact prompts, review loops, and real multi-agent usage from Simon Willison, Kent C. Dodds, Peter Steinberger, thdxr, and Addy Osmani.

### TOP SIGNAL

The clearest workflow convergence today: keep the top-tier model on **judgment**, not implementation. Simon Willison says telling Fable `For all coding tasks use your judgement to decide an appropriate lower power model and run that in a subagent` caused Claude Code to persist that instruction as project memory, route substantive work to Sonnet and trivial edits to Haiku, and reduce token burn; @anshnanda is using the same split more aggressively via `CLAUDE.md`, keeping Claude for planning/discussion and pushing coding, research, and other token-heavy work into Codex or subagents. Addy Osmani's autonomy framework points at the same durable lesson: **calibrated autonomy** wins because verification is still the constraint. [1, 2, 3]

> "Verification will always be the bottleneck." [3]

### TRY THIS

- **Route by task, not by model brand.** In Claude Code/Fable, try Simon's exact instruction: `For all coding tasks use your judgement to decide an appropriate lower power model and run that in a subagent.` Claude saved it to `~/.claude/projects/name-of-project/memory/delegate-coding-to-s`

Simon's split was Sonnet for substantive implementation, Haiku for trivial or mechanical edits, and the main high-power loop for design, auditing, review, and synthesis. If you want harder boundaries, @anshnanda puts this in `CLAUDE.md`: `You are primarily used to PLAN and DISCUSS various strategies that require critical thinking...` plus `ALL coding, discovery, implementation, research, and token-intensive tasks MUST happen using the use-codex skill`, then installs the use-codex skill. @anshnanda says that setup can cut Fable token usage by 90%. [1, 2]

- **Write a contract before any non-trivial handoff.** Addyo's pre-run template is portable across Claude Code, Codex, and anything else: define the *goal, scope, non-goals, tools/permissions, stopping condition, evidence, escalation path, and budget* up front. Use **Level 2** for bounded tasks where you stay nearby; only step up to **Level 3** when success can be measured and automated in plan → act → test → review loops. [3]
- **Triage big changes with a file-by-file summary — then demand evidence.** @thdxr's move: after a large change, ask the agent for a summary of what it changed in each file. He says weird stuff surfaces fast, and the best first-pass signal is *files + function signatures*; Kent says he's working the same way too. Just don't let that become **summary substitution**: Addyo explicitly warns that acceptance still needs the evidence packet — diff, tests, logs, screenshots, reviewer findings, risks, and gaps. Primeagen's counterpoint is the right gut check here: if your hands never get dirty, you still may not really know the state of the project. [4, 5, 3, 6]
- **Give agents real UI surface area when the task demands it.** Peter Steinberger's blunt advice: give the agent its own computer so it can do actual end-to-end testing — including clicking through OS alerts. And if Codex is weak on visual direction, try: `use imagegen to re-imagine this design and implement that`. This lines up with Addyo's note that goal-driven autonomy gets useful only when the environment is real enough and the stopping condition is measurable. [7, 8, 9, 3]

## WHAT SHIPPED

- **Fable orchestration + Composer 2.5, in a real migration.** Kent used Fable orchestrating Composer 2.5 to migrate his site from Fly.io to Cloudflare; the artifact to study is PR #813. In a separate note, he says Fable 5 orchestrating a swarm of subagents is strong at finding the critical path and splitting work for parallel execution. [10, 11]
- **Cursor cloud-agent coordination, not just one-shot prompting.** Kent describes one original Cursor cloud agent coordinating **6** child agents and sharing learnings across them — a concrete mainstream-tool example of manager/worker-style orchestration. [12, 3]

- **Claude Code vs Codex: today's feature map.** Addyo's comparison snapshot groups Claude Code around `/plan`, `/goal`, `/loop`, `/background`, `/batch`, `/code-review`, `/security-review`, plus subagents, hooks, checkpointing, and background sessions; Codex around Goal mode, worktrees, Automations, subagents, review panes, GitHub code review, hooks, sandboxing, Auto-review, and rerun. The writeup says it draws from analysis of ~400K Claude Code sessions, and separately points to teams already running hundreds or thousands of agents with continuous verification. [3]
- **Bug to watch: pi `edit` parameter injection reports.** Armin Ronacher says multiple users report Anthropic models adding extra tool parameters during `edit` operations in pi, but he hasn't been able to reproduce it; tracking issue: earendil-works/pi#6278. [13, 14]

## GO DEEPER

- **Simon Willison — "Fable's judgement".** Read this for the exact subagent-routing prompt and the memory file Claude created from it. [1]
- **kentcdodds.com PR #813.** A real migration artifact for studying how Fable-orchestrated Composer work looks on an actual site move. [10]
- **Addy Osmani — "Agentic Autonomy Levels".** Worth studying for the autonomy ladder, the anti-patterns, and the metrics that tell you whether your setup is actually working. [3]
- **use-codex skill.** If you copy @anshnanda's routing pattern, inspect the skill itself before wiring it into your own flow. [2]
- **pi issue #6278.** Relevant if you maintain tool-calling editors or want to track weird model/tool boundary failures early. [13, 14]

*Editorial take: the alpha today wasn't "more autonomy" — it was lower-power workers, tighter scopes, and harder proof loops, because verification is still the bottleneck and the slop problem is still real.* [1, 3, 6]

---

**Sources**


1. Fable's judgement
2. X post by @anshnanda
3. Agentic Autonomy Levels
4. X post by @thdxr
5. X post by @kentcdodds
6. X post by @ThePrimeagen
7. X post by @steipete
8. X post by @steipete
9. X post by @steipete

10. X post by @kentcdodds
11. X post by @kentcdodds
12. X post by @kentcdodds
13. X post by @mitsuhiko
14. X post by @mitsuhiko