# Route by Task, Gate by Verification

Coding Agents Alpha Tracker

2026-06-13

## Route by Task, Gate by Verification

*By Coding Agents Alpha Tracker • June 13, 2026*

Practitioners converged on a clear pattern today: use frontier models for planning and orchestration, route execution to cheaper models, and only let loops run unattended when they have hard verification or approval gates. Also covers new human-in-the-loop releases from Simon Willison, Codex reset controls, and open-source tools worth adding to your stack.

### TOP SIGNAL

- **The best practitioners are routing models by job, not betting on one model for the whole loop.** Cat Wu says Fable 5 is her default for careful production-codebase changes while Opus 4.8 still fits greenfield prototypes; Riley Brown used Fable 5 to spec a complex web→native iOS/Android conversion, then had Opus 4.8 implement it in one shot after 71 minutes; Theo says to explicitly tell Fable to use Opus or Sonnet for subagents because otherwise it tends to delegate to itself; and steipete's real-world comparison makes the economics obvious: similar feature, about $20 on deep^2 versus $350-$457 on Fable [1, 2, 3, 4].
- **The corollary:** don't drown the agent in instructions. DHH says overstuffed steering files can fill the context window with distractions and yield worse code, while human review and architectural judgment still matter even when agents write more of the code [5].

### TRY THIS

- **Spec with the strongest model; implement with the cheaper one.** Practical recipe:
  1. Give Fable 5 the hardest planning task: architecture, migration plan, or cross-platform spec.
  2. Hand that spec to Opus/GPT/Codex for implementation.

3. If the orchestrator is spawning subagents, explicitly tell it to use Opus or Sonnet underneath to control burn.
4. On review, ask it to justify edge cases and timing-sensitive code. Riley Brown used exactly this split for a web app → native iOS/Android conversion; Theo's tip is the cost control that makes it usable day to day; DHH's workflow is to challenge suspicious-looking code until the agent explains why it is necessary [2, 3, 5].

- **Prompt goals + constraints, not step-by-step tutorials.** Cat Wu's advice: give the model a higher-level problem, let it brainstorm, and state your constraints upfront; Fable is better at deducing intent from vague prompts and knows when to ask clarifying questions. DHH's warning pairs well with this: don't bloat steering files with instructions for things the model already knows [1, 5].

  > "If you try to tell an agent to do something in a certain way that it already kind of knew how to get right, you actually make it dumber." [5]

- **Add explicit approval checkpoints to tools.** Simon Willison's latest agent stack gives a clean pattern you can copy today:
  1. Let tools accept a `context` parameter.
  2. Call `await context.ask_user(...)` for yes/no, multiple choice (`options=[...]`), or free-text (`free_text=True`) input.
  3. Persist unanswered questions so the suspended run survives restarts.
  4. Re-run the tool from the top after answer replay, so ask *before* side effects. Backing this, LLM 0.32a3 adds `PauseChain`, unique `tool_call_id`s, and resume semantics for unresolved tool calls [6].
- **Only let loops run unattended when success is externally verifiable.** Good templates from today:
  - steipete lets Codex run for four days across multiple trees because the changes are end-to-end verifiable; the human mainly adds credit-card details and prunes bad ideas [7].
  - Theo has Mythos rank open PRs nightly and emit HTML plans that humans or other agents can inspect fast [3].
  - Armin says loops worked for a language port because the target was crisp: make the test suite pass; he avoids broader autonomous loops when QA/comprehension is fuzzy [8]. If you can't define the harness, keep the loop supervised—Kent C. Dodds describes an agent wiping production and its volume backups in nine seconds [9].

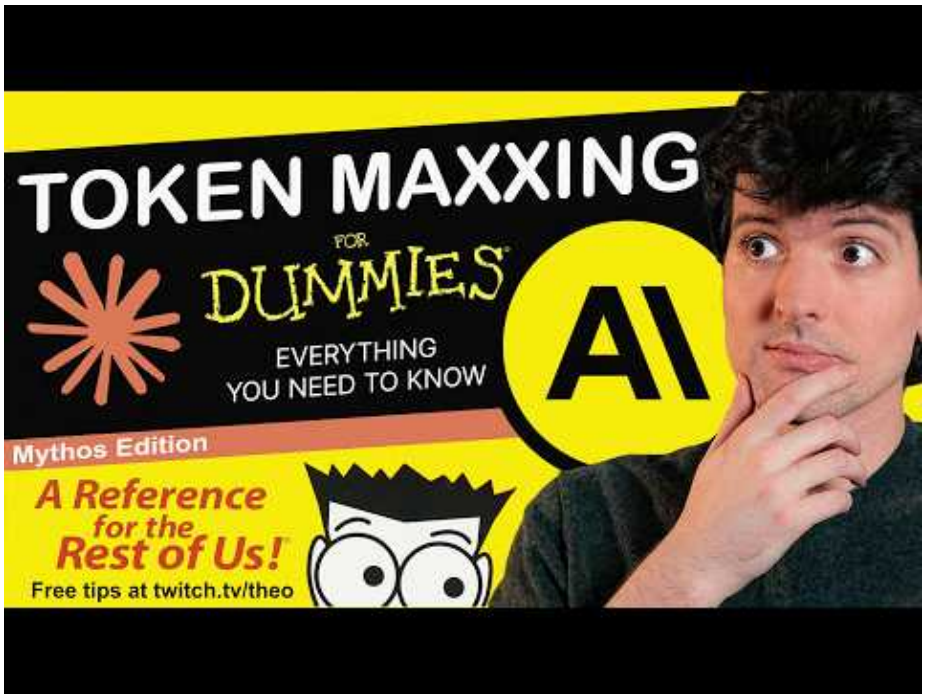
## WHAT SHIPPED

- **LLM 0.32a3** — tool-call plumbing for real human-in-the-loop agents: `llm_tool_call` injection, unique `tool_call_id`, `llm.PauseChain`, pause behavior for concurrent tool calls, resume-from-pending-tool-calls, and a fix for missing-tool async results [6].

- **datasette-agent 0.2a0** — `ask_user()` mid-execution plus a `save_query` tool that always requires human approval before storing SQL [6].
- **datasette-agent-edit 0.1a0** — new base plugin for agentic text editing, built around Claude-style `view`, `str_replace`, and `insert` tool patterns [6].
- **asyncinject 0.7** — Simon says Fable found bugs in the dependency and fixed them [6].
- **Pi 0.79.2** — `pi update` gets fable fixes and improved `/model` autocomplete [10].
- **OpenAI docs agent** — on developers.openai.com, the docs assistant answers product questions, links directly to relevant docs, and can generate a custom project guide you hand off to Codex [11, 12].
- **Codex reset controls** — Go/Plus/Pro/Business users can now save a rate-limit reset and apply it later; rollout starts with one free reset. Announcement: OpenAI on X [13, 14].
- **Local model signal** — Armin says Dwarf Star 4, a packaged quantized DeepSeek 4 Flash setup, is the first local model he has found viable for serious coding work, with roughly 450 tokens/s prefill and 25-26 tokens/s generation [8].
- **Open-source projects getting real pull** — Agent Skills packages a spec→plan→build→test→review→simplify→ship workflow and sits above 56k GitHub stars; Headroom wraps Claude/Cursor/Codex to compress context and show savings via `headroom perf`, with 24k+ stars; Last 30 Days turns recent internet discussion into agent-generated briefs and is above 40k stars [15].
- **Cost/guardrail reality check** — Fable's public spec is 1M context at $10/M input and $50/M output; Simon saw one debugging session price out around $12 and $110+ in a day, steipete saw $350-$457 on one feature and argues GPT can be 10-20x more token+cost effective for a similar outcome, and Riley says deeper frontier-LLM or security topics auto-route Fable sessions to Opus 4.8 [6, 4, 16, 17].

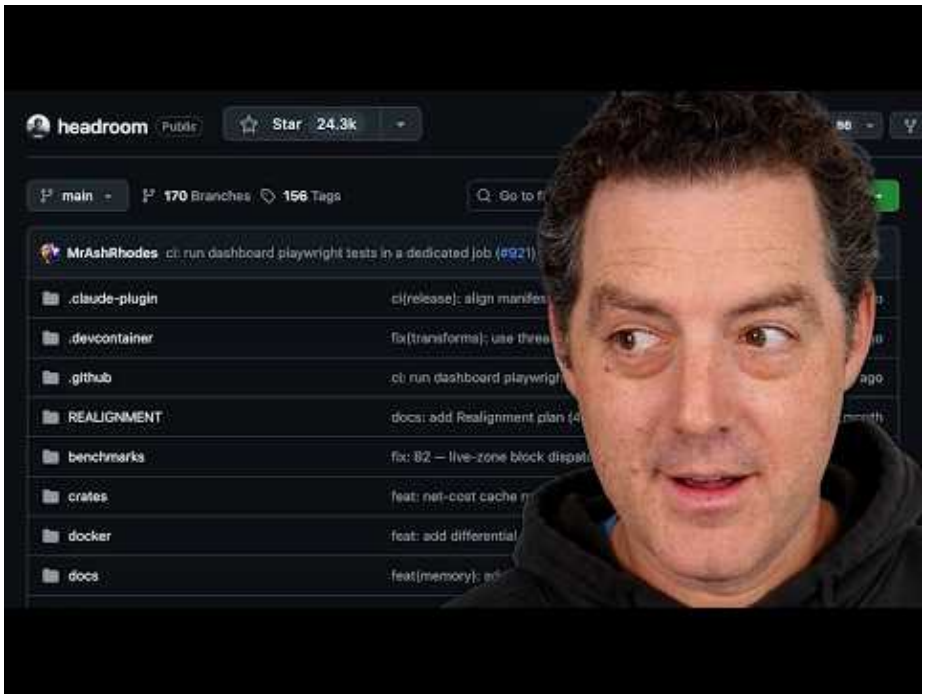
## GO DEEPER

- **25:16-29:14 — Theo on orchestration economics.** Useful clip on why Fable is great at coordinating judge/harvest/synthesis swarms, but should be told to use Opus/Sonnet subagents if you care about burn [3].




*Mythos is here, it's time to start tokenmaxxing (25:15)*

- **11:23-15:36 — Matthew Berman on Headroom.** Good hands-on demo of wrapping Claude Code, measuring savings with `headroom perf`, and mining failed sessions with `headroom learn` [15].




*You NEED to try these open-source AI projects RIGHT NOW (11:22)*

- **Study crabbox providers.** steipete says Codex is running *inside* crabbox while building crabbox, with browser/computer-use signups and long loops across multiple trees; valuable if you want a real self-hosted agent-control reference point [7].
- **Read the release notes for LLM 0.32a3 and datasette-agent 0.2a0.** This is one of the clearest public examples of how to build approval-aware tool calls instead of pretending full autonomy is safe [6].

*Editorial take: the practical edge is no longer "pick the best model"; it's route by task, define a verifier, and make approval pauses a first-class part of the loop.* [2, 3, 6, 8]

---

**Sources**

1. Claude → Fable 5 AI " " 1on1 Tech
2. X post by @rileybrown
3. Mythos is here, it's time to start tokenmaxxing
4. X post by @thorstenball
5. DHH: Basecamp 5, Vibe Coding, and the Future of Rails
6. Claude Fable is relentlessly proactive

7. X post by @steipete
8. State of Agentic Coding #7 with Armin and Ben
9. X post by @kentcdodds
10. X post by @mitsuhiko
11. X post by @OpenAIDevs
12. X post by @romainhuet
13. X post by @OpenAI
14. X post by @thsottiaux
15. You NEED to try these open-source AI projects RIGHT NOW
16. X post by @steipete
17. Claude Fable Will Change EVERYTHING (Here's Why)