Multi-Model Routing, Prompt-First Shipping, and Self-Updating Repo Docs

Coding Agents Alpha Tracker

2026-07-02

Multi-Model Routing, Prompt-First Shipping, and Self-Updating Repo Docs

By Coding Agents Alpha Tracker • July 2, 2026

The highest-signal workflows today were hybrid: route tasks between models, keep humans on architecture review, and give agents durable repo context. Also worth tracking: OpenWiki shipped, DeepAgents added RLM workflows, and Cursor updated its model lineup with Fable 5 and Kimi K2.7 Code.

TOP SIGNAL

The best coding-agent setups today looked **hybrid, routed, and heavily gated**—not autonomous. Theo says his Fable workflow improved end-to-end agent PR acceptance from roughly 50% rejected to 0% closed in a day by keeping Fable on `high`, routing implementation/computer-use/UI verification work to Codex/GPT-5.5, and codifying model priority rules in `CLAUDE.md` [1, 2]. DHH described the same pattern at 37signals: most Basecamp 5 fixes and feature upgrades now start with a prompt, but beta testing and senior review still decide what survives because working AI code can still damage architecture, performance, or security [3].

TRY THIS

- **Route by task, not by loyalty (Theo).** 1) Run Fable only on `high`. 2) Put model-priority rules in `CLAUDE.md`. 3) Teach Claude Code to call Codex/GPT-5.5 for implementation work. 4) Offload token-heavy computer use and codebase analysis to other models, then feed the results back to Fable. Theo says Codex still beats Claude for computer use, UI/UX verification, and well-specified execution; this setup also stayed under rate limits on the $200 plan [1, 2, 4].

- **Use a prompt-to-beta-to-review merge path (DHH, 37signals).** 1) Let designers, PMs, or juniors prompt the feature until it works. 2) Put it on a branch and beta with real data. 3) Get senior review on Ruby/JS before merge. 4) If the AI implementation is wrong, keep the idea and front-end work, then rewrite the code. DHH says this guardrail existed because raw AI output sometimes created technical debt or introduced architecture, performance, or security problems [3].
- **Give every coding agent a living repo manual (LangChain OpenWiki).** `npm install` -> `openwiki init` -> choose provider/model -> optionally add a LangSmith key for traces -> commit the generated `OpenWiki/` docs -> add a GitHub Action running `openwiki update` daily, weekly, or every 4 hours. OpenWiki will also create or update `agents.md/Claude.md` so agents know to consult the docs whenever they need codebase context [5].
- **Make verification machine-playable and visible (ThePrimeagen + Peter Steinberger).** Primeagen's pattern: add lots of asserts, build a JSON-playable version of the app or game, then let an LLM run 1,000 playthroughs to find bugs or crashes and auto-create Linear tickets [6]. Steinberger's lighter-weight variant: point Codex at user feedback, then let computer use attach before/after screenshots directly inside the PR when no GitHub API is available [7].

WHAT SHIPPED

- **OpenWiki** — new open-source LangChain agent for codebase docs. It generates repo docs, auto-updates them as the codebase evolves, does Q&A over docs/codebase, and ships here: langchain-ai/openwiki [5, 8]. The output includes a `QuickStart.md` index plus architecture, CLI, business-logic, and git-history files built for coding agents to consume [5].
- **DeepAgents + RLMs** — LangChain showed RLM support via code interpreter middleware. The main agent gets a `task` function so it can orchestrate subagents in code, and the `workflow` keyword triggers the mode in Decode [9]. On the Oolong long-context task, RLM-enabled DeepAgent did much better at 128k context than plain DeepAgent, which often gave up with "I can't answer"-style responses; tradeoff is more latency and higher token cost [9].
- **Decode + GLM 5.2** — install is "copy the script from the DeepAgents docs site, then run `decode`"; init supports Fireworks + `GLM5P2`, and `auth` can wire Tavily or LangSmith tracing [10]. Useful built-ins: `threads`, `offload`, `mcp`; GLM 5.2 was presented at 81 on terminal-bench and 79 tokens/sec, with a demo building an LLM chat app end to end [10].
- **Cursor's model board moved again** — Claude Fable 5 is back in Cursor, Cursor says it leads CursorBench, and Cursor also says it's the

most expensive per task; full comparisons: cursor.com/evals [11, 12]. Kimi K2.7 Code also launched in Cursor, with K2.5 scheduled to go away Friday; Cursor shared evals comparing K2.7 against GLM 5.2 [13, 14, 15].

- **Real adoption signal: LangSmith at Pendo** — Pendo says LangSmith catches 60% of agent failures before customers see them. Their daily loop is simple and worth stealing: open the trace dashboard, find the gap between customer needs and current agent behavior, then build new eval sets [16].

GO DEEPER

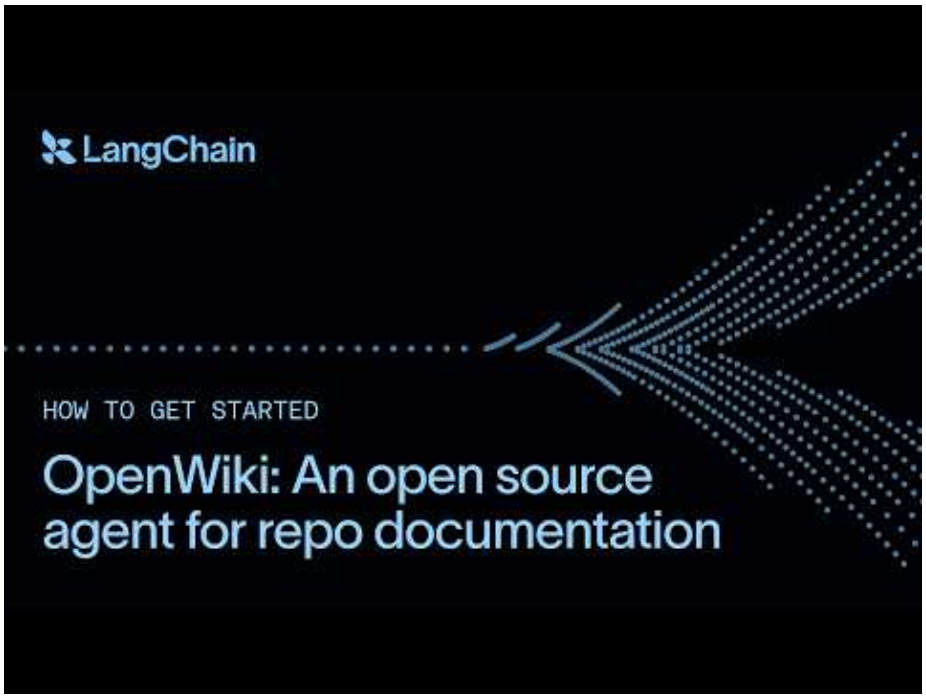
- **2:11-2:53 — DHH on prompt-first development.** Good clip if you want evidence that AI is now in the main path for a real software team, not just sidecar autocomplete [3].

AI Challenges in Software Development – REWORK (2:11)

- **8:06-9:09 — Basecamp's merge gate.** Best short explanation today of why "it works" is not a sufficient acceptance test for AI-generated code [3].

AI Challenges in Software Development – REWORK (8:05)

- **4:50-5:59 — OpenWiki wiring into `agents.md` / `Claude.md`.** Watch this if you want agents to discover repo docs automatically instead of re-explaining the codebase in every session [5].



Introducing OpenWiki, an open source agent for repo documentation (4:50)

- **4:15-5:17 — Decode -> LangSmith tracing.** Nice walkthrough of turn-by-turn traces, token usage, and tool-call inspection for long-running coding sessions [10].

GLM 5.2 + dcode: Frontier Coding with Open Models (4:15)

- **Study next:** langchain-ai/openwiki for self-updating agent docs [8]; OpenClaw PR 98452 for Codex-driven UI improvement and screenshot evidence inside a real PR [7].

Editorial take: today's alpha wasn't "pick one winner model" — it was ***route models by task, give them durable context, and force proof before merge****.* [2, 5, 6, 3]

Sources

1. X post by @theo
2. X post by @theo
3. AI Challenges in Software Development – REWORK
4. X post by @theo
5. Introducing OpenWiki, an open source agent for repo documentation
6. X post by @ThePrimeagen
7. X post by @steipete
8. X post by @LangChain
9. How to use RLMs in Deep Agents
10. GLM 5.2 + dcode: Frontier Coding with Open Models
11. X post by @cursor_ai
12. X post by @cursor_ai

13. X post by @jediahkatz
14. X post by @jediahkatz
15. X post by @leerob
16. X post by @LangChain