The Agent Harness Is Becoming Self-Improving

Coding Agents Alpha Tracker

2026-08-16

The Agent Harness Is Becoming Self-Improving

By Coding Agents Alpha Tracker • August 16, 2026

Exo's rollback-protected recursive harness and Flue 2's dynamic Agent Hooks point to a shift from static prompts toward evolvable, testable coding-agent control planes.

TOP SIGNAL

The harness is becoming the agent. Fred Schott's Flue 2 makes an agent a JavaScript function that re-renders before every model call; its TypeScript hooks manage state and lifecycle, and attach skills, tools, and subagents dynamically—for example, adding account-management access only after a support bot verifies the user. [1] Alex Krentsel's Exo takes the harder route: policy lives in a stateless executor, history/secrets/snapshots in a protected harness, commands in a sandbox, and a guardian can rebuild the executor and roll it back if the new version breaks. [2]

The practical consequence is not "give the model more autonomy"; it is "give autonomy explicit boundaries, snapshots, and evals." Krentsel argues that architectural guarantees beat prompt-level rules for properties such as never deleting history, and warns that cost optimization without evals can reward-hack by simply stopping work. [2]

TRY THIS

- **Close the context-cost loop.** Add per-message cost annotations to the conversation log. Ask the agent to inspect its last expensive call, narrow context to the active conversation or thread, observe and test the change, and commit the improvement only after a functional eval. Exo reports taking a Discord call that cost 16¢ to roughly 96% cheaper this way; its own warning is the important part—without an eval, "do nothing" is the cheapest possible optimization. [2]

- **Separate what can change from what must be protected.** Keep the executor's policy stateless; store conversation history, secrets, artifacts, and snapshots in the host-side harness; run shell/filesystem actions in an isolated sandbox. If the agent edits its own executor, let a guardian rebuild it for one step and automatically roll back on failure. Keep secrets out of the tool-visible container and inject them only into the model call. [2]
- **Gate capabilities by workflow state instead of dumping every tool into context.** In Flue 2, model the agent as a TypeScript/JavaScript function that re-renders before each model call. Use lifecycle/state hooks to add `useTool()` or `useSubagent()` only when the task warrants it; the concrete pattern in the launch discussion is verifying a support user before attaching an account-management tool. [1]
- **Budget both tokenizer and context.** Compare actual token counts and successful outcomes on your own corpus, not just advertised dollars per million: in one small mixed-text test, GPT-5.6 Sol used 766 tokens versus an estimated 1,170 for Claude Opus 5—about 34.5% fewer—and the recommendation is to measure price per successful outcome yourself. [3] For local high-reasoning models, set the context length deliberately: Simon Willison's Qwen 3.8 27B "extra high" run hit the default context limit, then succeeded after he increased it. [4, 5, 6]

WHAT SHIPPED

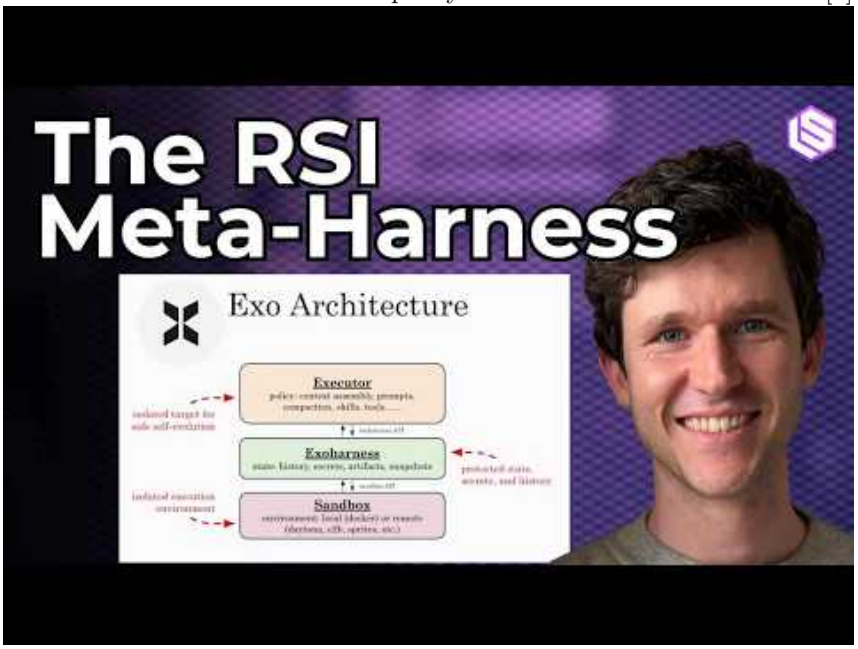
- **Flue 2 reached its first stable release.** The React-style Agent Hooks API ships 16 built-in hooks—including `useSkill()`, `useTool()`, and `useSubagent()`—plus custom hooks, and is built on the minimal open-source Pi harness. [1]
- **Multi-agents v2 added cross-model delegation.** Builder @pvncher says models can now delegate to any supported model, including Luna, after reliability work; @thsottiaux frames the intended pattern as Sol managing a fleet of Luna agents. That is a useful routing primitive, but the posts provide no quality benchmark. [7, 8]
- **DHH's plan-driven code-model comparison widened.** DeepSeek Pro V4 Max reportedly completed the TerminalTextEffects Rust-rewrite challenge in 2h30m for $23, versus roughly $550 for Fable in 45 minutes and $55 for Grok 4.6 in 1.5 hours; DeepSeek V4 Flash and GPT Luna failed. [9, 10] Every implementation used the plan Fable wrote, so this is evidence about execution cost and plan-following—not independent project planning. [11]
- **Exo has a real production signal, not just an architecture pitch.** Krentsel says the harness and agents built on it are running in production at Braintrust; the project offers a one-line install, Discord/IRC/WhatsApp adapters, and Discord voice mode, though the latter is still a pipeline

cascade rather than an interactive model. Study the EXO repository. [2]

- **Antigravity's Gemini 3.7 Flash integration targets cross-platform UI generation.** Antigravity says the model can build complete native authentication screens across SwiftUI, React Native, Jetpack Compose, and Flutter; the update is available by download or upgrade. Treat this as a vendor demo claim—no benchmark is supplied in the announcement. [12, 13]

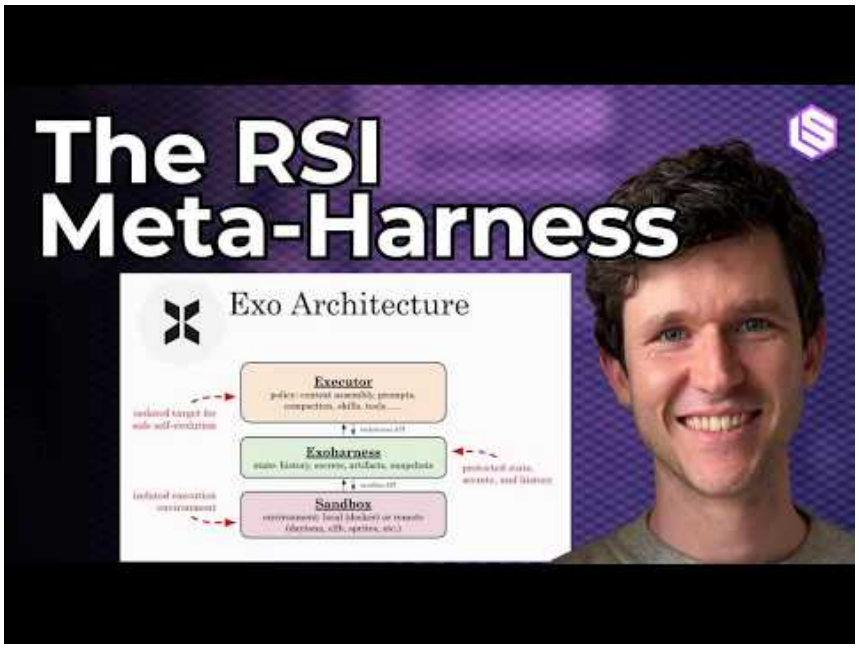
GO DEEPER

- **Exo: "Harnesses should see their own code and logs" — Alex Krentsel** — Start with the executor/harness/sandbox split and guardian rollback, then watch the Discord example for an agent that rewrites its own context policy and measures the result. [2]



Exo: Harnesses should see their own code and logs — Alex Krentsel (16:18)

- **Exo cost-aware self-optimization segment** — The useful implementation detail is per-message cost logging, thread-scoped context, and the reward-hacking caveat that makes evals mandatory. [2]



Exo: Harnesses should see their own code and logs — Alex Krentsel (39:09)

- **Flue 2 launch post** — Study the Agent Hooks model and the "no agent without a harness" thesis; the design is a practical counterexample to file-based routing and static tool configuration. [1]
- **ttfx plan** — Reuse the fixed plan as a controlled artifact when comparing models; it keeps planning quality separate from execution speed, cost, and reliability. [10, 11]

Editorial take: Models are becoming replaceable components; the durable edge is a harness that can change its policy without losing state, leaking secrets, or gaming its own objective. [2]

Sources

1. React for Agents: Astro Creator Brings Hooks to his Meta-Harness, Flue
2. Exo: Harnesses should see their own code and logs — Alex Krentsel
3. X post by @thsottiaux
4. X post by @simonw
5. X post by @simonw
6. X post by @simonw
7. X post by @pvncher
8. X post by @thsottiaux
9. X post by @dhh

10. X post by @dhh
11. X post by @dhh
12. X post by @antigravity
13. X post by @antigravity