Jev's Control-Plane Pattern Gets a Real Harness

Coding Agents Alpha Tracker

2026-09-19

Jev's Control-Plane Pattern Gets a Real Harness

By Coding Agents Alpha Tracker • September 19, 2026

Jev's launch produced a clearer coding-agent architecture: fast typed decisions and state machines around slower generative models, reinforced by portable instructions and sandboxed execution.

TOP SIGNAL

The useful Jev signal today is an implementation pattern, not a replacement-model claim: put a typed control plane in front of the coding model. Jev returns a predefined `choice`, `score`, or `null` decision; Riley Brown used that contract to route a simple request to Nano and an app-architecture request to Sonnet while the downstream model handled files and code. [1]

Matthew Berman shows the same split from another angle: Astra/Codex built a simulated world, Jev made the characters' decisions, and Jev sits between the prompt and the model that actually answers it rather than coding from scratch. [2] Riley's 500-email demo finished in roughly 12–13 seconds; his comparison of 0.4 seconds/$0.00004 per Jev decision versus roughly 10 seconds/$0.03 for a traditional LLM is vendor-reported economics, not a coding benchmark. [1]

TRY THIS

- **Route the hot path by schema.** Define a finite `choice` set, a score scale, or a binary `null` question; ask several questions over the same state, then hand the resulting task to the appropriate generative model. Kody's Jev integration maps directly to ticket triage, urgency/escalation, "enough detail to file an issue," and ready-to-ship scoring, with `choice`, `score`, and `null` questions in one call. [3] For a quick prototype, use Riley's prompt shape: `Create an app that uses Jev. Use Jev. Look up the docs.` Jev's 64,000-token input limit makes payload discipline part of the design. [1]

- **Make UI autonomy hierarchical, not flat.** In ThePrimeTime's Bellatro prototype, start with a structured "god view," ask one typed question such as `What is your next move in Bellatro?`, expose only the actions relevant to the current screen, then decompose `play hand` into a lower-level card-selection loop. [4] Use `name/target` payloads and the `enabled` flag instead of raw cursor clicks; log the executor, and remove stale fields such as a misleading `reason` value. [4] Prime found the raw snapshot consumed roughly 20,000–21,000 input tokens, then reduced it to hand, score, plays/discards, jokers, tarot cards, chips needed, and hand values. [4]
- **Make capabilities portable, not sessions.** Riley Brown is centralizing skills, plugins, and keys so he can switch among Codex, GrokBot, Claude Code, Muse, and other platforms; he later says his agents share skills, plugins, and memory, with model switching even in an iMessage agent. Treat this as a portability experiment rather than a production benchmark, but copy the direction: one versioned capability layer with thin host adapters. [5, 6, 7]
- **Trigger outside the chat and measure each lane.** Kody's subscriptions support platform-level and custom events, which Kent C. Dodds uses to wake a bot from email and Discord. Pair that event layer with Ben Tossell's harness/token tracker, which splits usage by agent and model, before tuning prompts or adding more autonomy. [8, 9]

WHAT SHIPPED

- **Claude Code 2.1.277 adds `AGENTS.md` fallback.** If a folder has no `CLAUDE.md`, Claude Code now checks and uses `AGENTS.md`; the behavior is toggleable in `/config`. It is implemented as a built-in Claude Code mod, with custom project-instruction mods planned. [10, 11] Simon Willison says this removes his one-line `CLAUDE.md` wrapper workaround, while Romain Huet calls it ecosystem convergence around a shared standard. [12, 13] Study the mods source, especially the `AGENTS.md` implementation. [14]
- **Jev is now exposed directly in Kody Koala.** The integration is aimed at fast typed decisions over shared state, including classifying mixed tickets, deciding whether a PR is ready to ship, and assigning a rough quality score. LangChain says Jev reports up to 200× faster inference and 400× lower cost than comparable LLMs on classification tasks; treat that as a reported model claim, not an independent benchmark. [3, 15]
- **OpenClaw adds a local-to-sandbox handoff.** In the latest OC, ask the agent to `Run this [web app] in crabbox and show me [vnc / a portal]`; the flow now works when development starts locally, across Linux, macOS, and Windows boxes, with CUA support as well. [16, 17, 18] Steinberger also describes a Discord-connected `roboclaw` team server that

tracks current and past sessions, while a collaborator cleans ("deslops") sessions before the PR lands. [19, 20]

- **AgenticLinux packages the agent workstation as an immutable system.** The new bootc desktop ships with Docker Engine, Docker Sandboxes, `llmman`, and OpenClaw; its root is read-only, updates are atomic from Docker Hub, and rollback is built in. The GitHub repo is worth studying for the deployment boundary around local agents. [21]
- **API-tooling architecture is up for revision.** Armin Ronacher proposes a more direct MCP shape—codemode plus OpenAPI plus RAG over OpenAPI documentation—pointing to the OpenAPI-only Radius skill as evidence and arguing that MCP can be layered on top but was not designed for this boundary. Separately, `gog` now has an MCP server. [22, 23, 24]

GO DEEPER

- **Riley Brown — JEV: How It Works and What You Can Build:** watch the model-router and `choice/score/null` walkthrough; it is the cleanest explanation of why the decision layer belongs outside the coding model. [1]

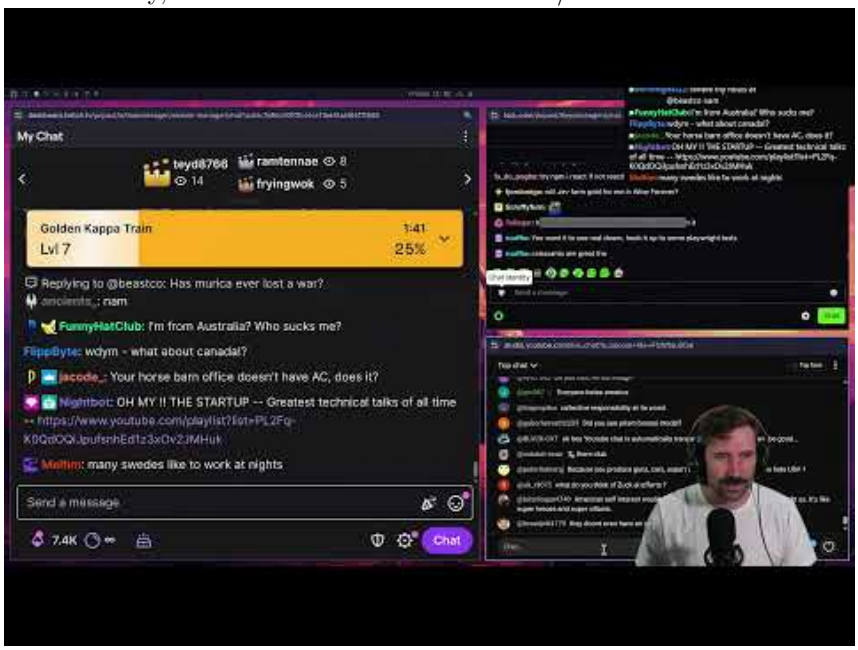
JEV: How It Works and What You Can Build (1:47)

- **Matthew Berman — We need to talk about Jev...:** the hybrid-world demo makes the boundary concrete—Astra/Codex builds the environment, Jev handles repeated in-loop decisions, and a router chooses where

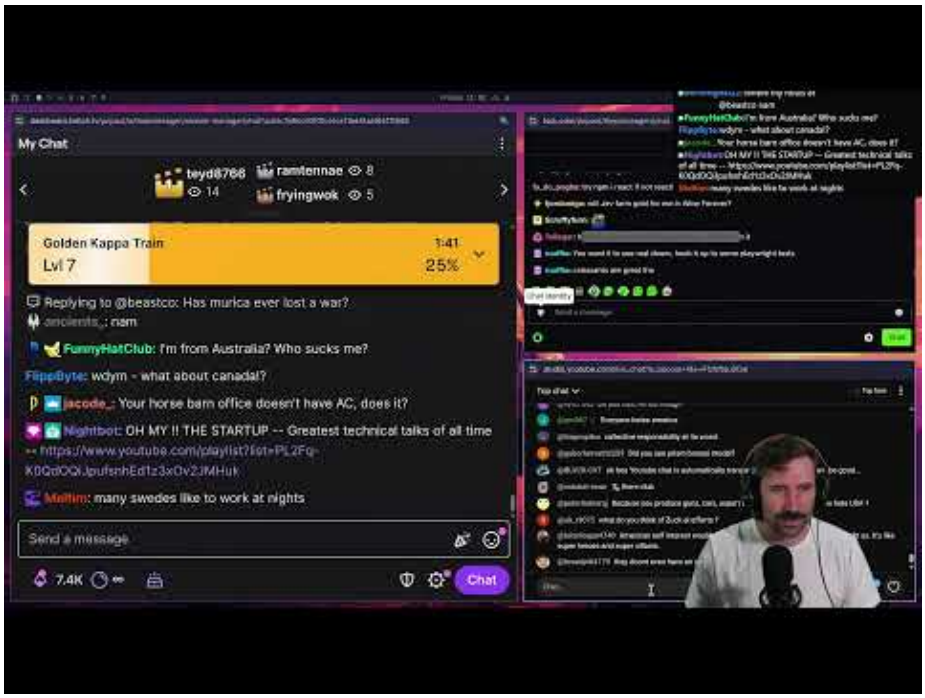
each request goes. [2]

We need to talk about Jev… (6:28)

- **ThePrimeTime — TRYING JEV: The new STYLE of AI!!!:** this is the useful teardown rather than a polished demo: state reduction, executor bugs, action validity, and the eventual behavioral-tree/state-machine design. [4]



TRYING JEV : The new STYLE of AI!!! (119:05)



TRYING JEV : The new STYLE of AI!!! *(135:48)*

Editorial take: The frontier model should spend its budget on hard reasoning; the harness should own routing, state reduction, action validity, portable instructions, and sandbox handoff. [1, 4, 16]

Sources

1. JEV: How It Works and What You Can Build
2. We need to talk about Jev…
3. X post by @kentcdodds
4. TRYING JEV : The new STYLE of AI!!!
5. X post by @rileybrown
6. X post by @rileybrown
7. X post by @rileybrown
8. X post by @kentcdodds
9. X post by @bentossell
10. X post by @trq212
11. X post by @trq212
12. X post by @simonw
13. X post by @romainhuet
14. Quoting Thariq Shihipar
15. X post by @LangChain

16. X post by @steipete
17. X post by @steipete
18. X post by @steipete
19. X post by @steipete
20. X post by @steipete
21. X post by @ericcurtin17
22. X post by @mitsuhiko
23. X post by @mitsuhiko
24. X post by @steipete