Coding-Agent Gains Are Moving Into the Harness

Coding Agents Alpha Tracker

2026-08-23

Coding-Agent Gains Are Moving Into the Harness

By Coding Agents Alpha Tracker • August 23, 2026

A 23.8-point harness spread makes context, compaction, permissions, and human attention first-class coding-agent surfaces, while practitioner reports show how to delegate and parallelize real work.

TOP SIGNAL

Benchmark the harness, not only the model. A new Latent Space analysis reports the same model scoring **52.4–76.2 on the same 106 tasks** across different harnesses—a **23.8-point spread**—and GPT-5.6 Sol's ARC-AGI-3 score rising from **13.3% to 38.3%** through retained reasoning and compaction alone. [1]

The practical consequence: treat context, memory/compaction, tools, permissions, and guardrails as model-adjacent engineering surfaces, not wrapper polish. The analysis argues that as models absorb more of the scaffold, the remaining product is an "attention-interface" governing when an agent interrupts, what it can decide alone, and what needs approval. [1]

TRY THIS

- **Run an async, bounded overnight loop.** DHH gave Omabot a HEY account and beta CLI, kicked off work, and woke up to an email describing what it had completed; he expected to ship the result to edge that day. [2, 3] Replicate the pattern on a reversible task: provide a scoped account/CLI, define the deliverable and completion channel, let the agent work asynchronously, then review the output before shipping.
- **Use one painful surface as a three-prompt smoke test.** Theo first said he could build something better than the Plex app in under five prompts, then followed up with "3 prompts," streaming from his NAS over

Tailscale. [4, 5] Pick one irritating personal workflow, preserve the existing data path, ask for a working end-to-end slice, and cap the experiment before architecture work takes over.

- **Parallelize only after the spec is legible.** Matt Pocock's in-progress `/implement-spec` skill takes a spec and tickets, researches the codebase in a subagent, implements tickets in maximally concurrent subagents, reviews against the spec, and cleans up worktrees. [6] Kent C. Dodds supplies the needed counterweight: being overly prescriptive can undo optimizations made by the model and harness. Use the phases as contracts and outcomes, not as a script for every subagent move. [7]
- **Make verification behavioral—and do not accept the first "impossible."** Simon Willison frames productive agent use as confidently instructing the change and then confidently verifying it, rather than assuming line-by-line eyeballing is the best validation method. [8] Linus Torvalds describes the concrete debugging version: after an AI repeatedly declared a kernel problem unsolvable, he pushed it to keep adding debug code and analyzing faithfully, then let it write the commit message. Ask for instrumentation and evidence before treating a refusal as a conclusion. [9]

WHAT SHIPPED

- `llm 0.33` — Simon Willison's CLI upgraded to the OpenAI Python library 3.x and `httpx2`; embeddings now accept per-call `--key`/`key=` values. [10] Repeated `-t/--template` flags now compose saved model configuration and prompts, including this directly usable pattern:

  ```
  llm -m gpt-5.6-luna -o reasoning_effort high --save lhigh
  llm "Generate an SVG of a pelican riding a bicycle" --save pelican
  llm -t lhigh -t pelican
  ```

  Reasoning-capable Responses API models also expose `reasoning_summary=auto|concise|detailed`. [10]
- **Codex CLI startup was rebuilt for latency.** Charlie Marsh says the latest release redesigned the lifecycle to make `codex` startup instant, roughly **25× faster** and immediately responsive. [11]
- **Kody Koala is positioning itself as a cross-agent runtime.** Kent C. Dodds says it augments, rather than competes with, tools including Cursor, Codex, Claude, OpenCode, OpenClaw, and Devin, with work portable across them. [12]
- **Vision is becoming a practitioner selection bar.** Theo's explicit view—not benchmark data—is that a code model unable to inspect a screenshot of an issue is "wholly disinteresting"; he calls vision a base-level requirement. [13]

GO DEEPER

- **DHH's Linux/Omarchy video — 10:20.** DHH linked this exact timestamp while arguing that "Linux is fucking happening"; use it as the visual companion to the async Omabot workflow above. [14]
- **The Evolution of the Agent Harness.** Read the Harness-Bench comparison alongside the proposed attention-interface: the useful question is not merely which model wins, but which context, compaction, permission, and approval design lets the same model do reliable work. [1]
- **Armin Ronacher — Fast, Hard Code.** Ronacher argues that agents reduce the friction of unfamiliar languages, points to `pi-autoresearch` for agent-directed performance work, and notes LLM-assisted projects in "hard" languages such as Zig. The useful caveat is his own: domain knowledge still helps. [15]
- **Study the `implement-spec` skill.** It is a compact reference for turning a spec into codebase research, concurrent ticket execution, a spec review, and worktree cleanup—while the Kent warning above argues for leaving the harness room to optimize. [6, 7]

Editorial take: The frontier is shifting from giving agents more autonomy to engineering the boundary around them—preserve context, delegate asynchronously, parallelize bounded work, and spend scarce human attention only where the agent cannot decide alone. [1, 2]

Sources

1. The Evolution of the Agent Harness
2. X post by @dhh
3. X post by @dhh
4. X post by @theo
5. X post by @theo
6. X post by @mattpocockuk
7. X post by @kentcdodds
8. More than just code review
9. Quoting Linus Torvalds
10. llm 0.33
11. X post by @charliermarsh
12. X post by @kentcdodds
13. X post by @theo
14. X post by @dhh
15. Fast and Hard Code