Proactive Agent Loops and Auto-Review Enter the Workflow

Coding Agents Alpha Tracker

2026-06-12

Proactive Agent Loops and Auto-Review Enter the Workflow

By Coding Agents Alpha Tracker • June 12, 2026

Today's useful signal is the operational layer around coding agents: proactive debug loops, external judge agents, auto-review defaults, and transcript-level evals. This brief pulls together the most actionable workflows, tool updates, and clips worth studying.

TOP SIGNAL

- **Proactive loops are the real step change.** Simon Willison's firsthand Fable transcript is the clearest proof: from a screenshot and a one-line goal, the agent stood up a local server, scratch HTML, browser automation, DOM probes, and screenshot capture to isolate a two-line CSS fix—and Willison says that same relentless proactivity is exactly why unsandboxed agents are a security risk if prompt-injected [1]. Cursor's multi-agent practitioners described the same broader pattern at scale: plan-first swarms, reminder loops, and separate judge agents that keep long-running tasks from stopping early [2].

TRY THIS

- **Steal Simon Willison's browser-debugging escalation ladder.** Start with a screenshot and a one-line objective. If the bug does not reproduce in Playwright across Chrome/Firefox/WebKit, have the agent: run the local dev server with fake env vars; write a scratch HTML page that isolates the behavior; open real Safari/Firefox; use `uv run --with pyobjc-framework-Quartz` plus `screencapture -x -o -l` to capture real browser windows; inject JS to trigger hidden UI flows; and post DOM

measurements back to `127.0.0.1:9999` via a tiny CORS `http.server` so the agent can read the results from disk [1].

```
Write a report in /tmp/automation-report.md where you
note down all of the tricks you have used in this
session to test against real browsers on my computer,
include runnable code examples
```
[1]

- **Run a plan-first swarm, not a prompt pile.** In Cursor's multi-agent discussion, the advice was to spend most of the session in plan mode until you have a detailed markdown spec, then let a main agent delegate to named sub-agents via messaging scripts, with reminder loops that keep overnight runs anchored on explicit verification steps. For long runs, pass data across compaction cycles by reference instead of summaries [2]. If you push toward bigger fleets, don't assume GitHub is your control plane: swyx's summary from ai.engineer says runtime and triggers are mostly solved, but coordination is not, and GitHub gets too noisy for hundreds of parallel PRs—use a CLI or messaging gateway instead [3].
- **Split build vs. review across model families.** A concrete pattern from Cursor's panel: use Claude/Opus 4.7 for implementation or design, keep a GPT-5.5 worker around for review and functional verification, and run several rounds of "thermonuclear review" so different models catch different failure modes [2]. Simon Willison used the same basic routing in a fresh side project: Claude Fable 5 for the plan, GPT-5.5 xhigh in Codex Desktop for the implementation of a custom Datasette extras explorer [4].
- **Read trajectories like code review, not scoreboard output.** swyx's Frontier Code discussion argues for hard, real-world tasks plus maintainer-labeled trajectories and rubrics for mergeability, not just passing tests; the concrete checks are simple: did the agent touch the right files, write reasonable tests, and respect style/lint guidelines [5, 6]. Sourcegraph's 1,281-run study points the same way: failures in large repos show up as repeatable patterns tied to different infra fixes, not random bad luck [7].

WHAT SHIPPED

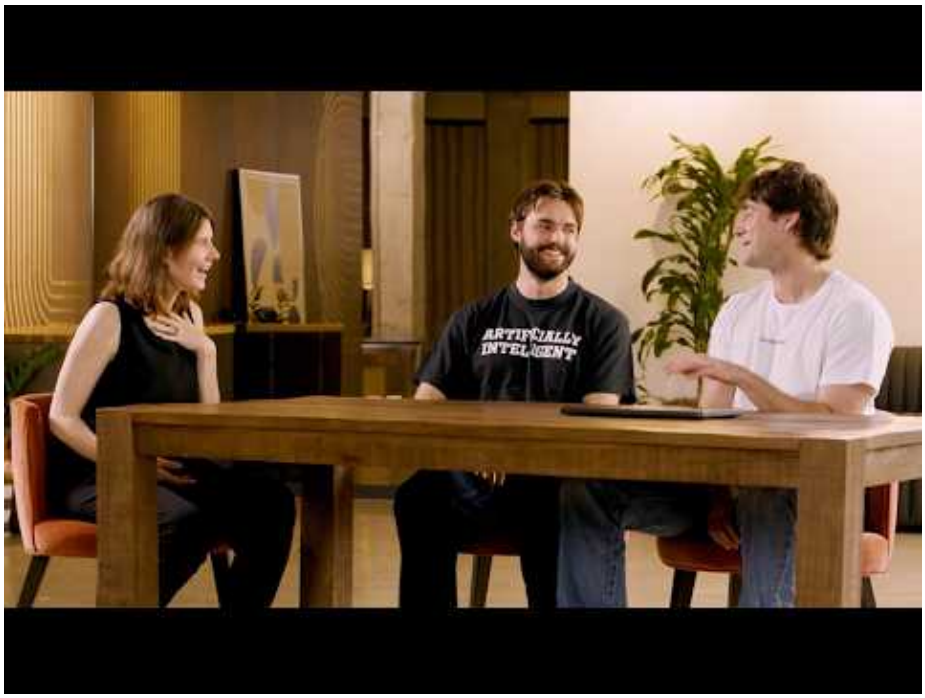
- **Cursor Auto-review** — now default for new users; a classifier subagent evaluates actions in context and decides whether to allow, block, or ask for approval. Cursor says evals show **97% accuracy**, with misses mostly near ambiguous edges. Details: cursor.com/blog/agent-autonomy-auto-review [8, 9].
- **Google's Anti Gravity harness, per Logan Kilpatrick** — framed as a shared IDE/web/CLI/SDK + Gemini API managed-agent layer, with the same harness powering Search, Gemini app, Cloud, and AI Studio; use-case variants appear to share roughly **80%** of the base harness [10]. Usage signals from Logan: Google teams used it to ship mobile and macOS apps

faster internally, AI Studio saw roughly **350k Android apps** built since last week, and about **20%** of early apps were games [10].

- **Gemini coding-model note** — Logan says **Gemini 3.5 Flash** is better at coding than any prior Pro model Google released, and credits the jump to post-training gains rather than a bigger base model [10].
- **Sourcegraph's large-repo postmortem** — **1,281** agent runs across **40+** open-source repos surfaced five repeatable failure patterns, each pointing to a different infra fix. Read: Why coding agents fail in large codebases (and what to do about it) [7].
- **Open-source maintainer loop from Peter Steinberger** — his Codex setup wakes every five minutes, steers work to threads, and combines an orchestrator with triage/autoreview/computer-use skills so some changes can land autonomously. Study the skill files: maintainer-orchestrator and github-project-triage [11].
- **Fresh small-project proof of routing** — Simon Willison shipped a custom extras API explorer with Fable 5 handling the plan and GPT-5.5 xhigh handling the implementation; separately, he says Fable spotted and fixed bugs in `asyncinject 0.7` [4, 12].

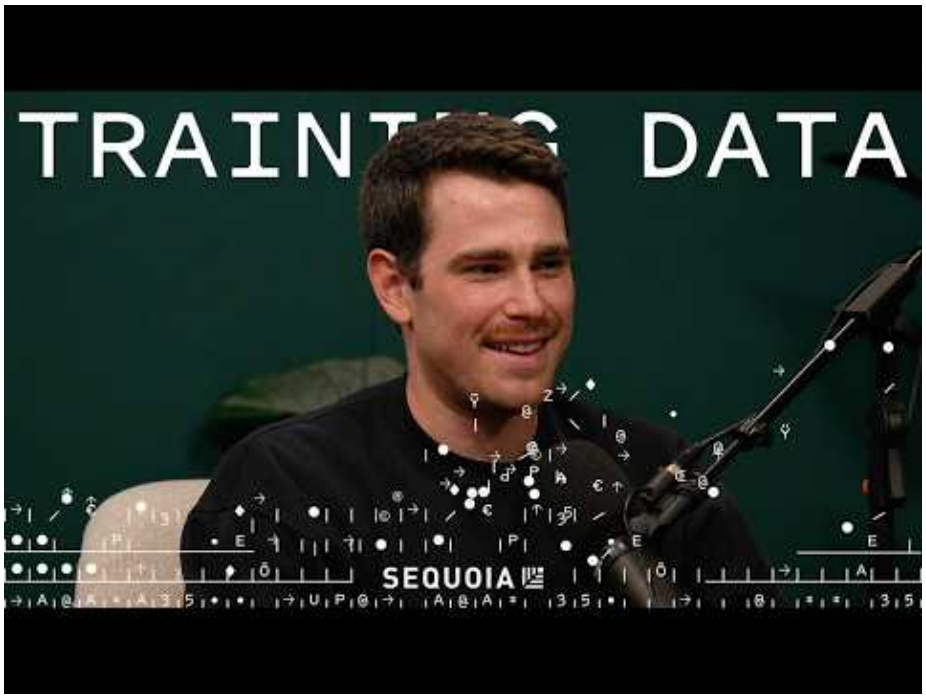
GO DEEPER

- **Cursor, 5:19-5:54 — separate judge agents.** Best short explanation of why a worker agent needs an external checker to stop false-done states before they spread [2].



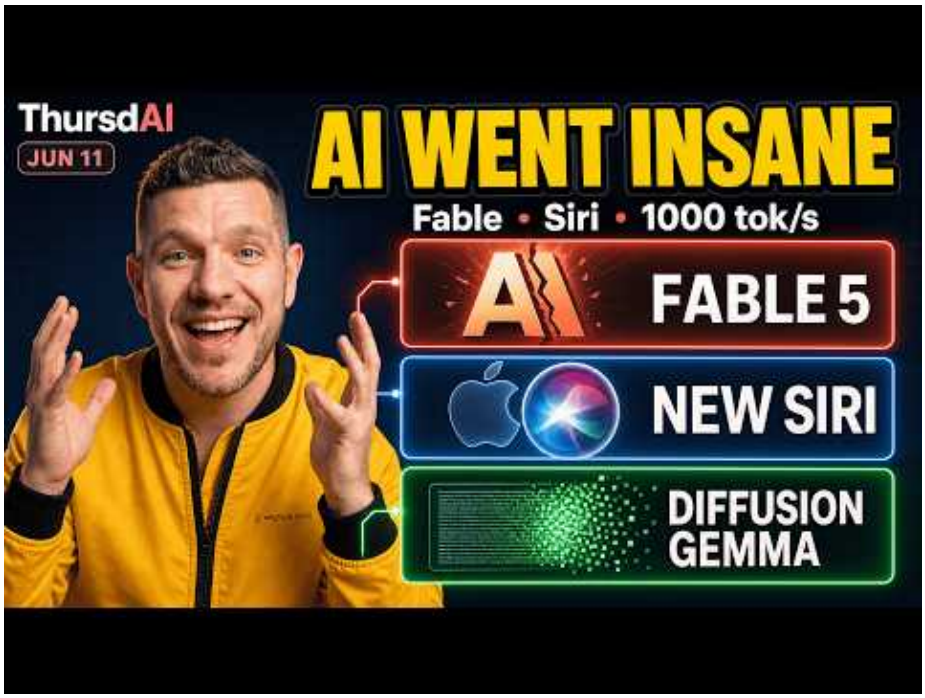
Running 128 Coding Agents at Once (5:19)

- **Google/Logan, 38:39-40:30 — the model eats the harness.** Useful mental model for why today's bespoke agent scaffolding keeps getting upstreamed into model behavior, and where the next edge moves [10].



Google DeepMind's Logan Kilpatrick: Why the Model Eats the Harness (38:38)

- **Frontier Code, 1:01:05-1:05:55 — trajectory reading party.** Best clip today on how to judge agent work beyond pass/fail: wrong-file touches, reward hacking, weak tests, and mergeability [6].



Fable 5 controversy, new Siri with computer use, Gemini Live Translate under 500ms | Jun 11 (61:05)

- **Study Simon Willison's full terminal transcript.** It's one of the best public artifacts right now for seeing an agent improvise across Playwright, Safari, DOM instrumentation, screenshots, and local servers in a real debugging task [1].
- **Study steipete's skill files.** If you want a concrete maintainer loop instead of general agent talk, the orchestrator and triage skills show how he structures autonomous repo maintenance around threads and review gates [11]. maintainer-orchestrator · github-project-triage

Editorial take: the edge is moving from raw codegen to agent control systems—loops, judges, artifact capture, and transcript review are what make longer autonomy usable.

Sources

1. Claude Fable is relentlessly proactive
2. Running 128 Coding Agents at Once
3. X post by @aiDotEngineer
4. datasette 1.0a33
5. LIVE - Fable/Mythos 5 is HERE , SIRI AI finally?, Frontier code w/ Swyx & more AI news

6. Fable 5 controversy, new Siri with computer use, Gemini Live Translate under 500ms | Jun 11
7. X post by @Sourcegraph
8. X post by @cursor_ai
9. X post by @cursor_ai
10. Google DeepMind's Logan Kilpatrick: Why the Model Eats the Harness
11. X post by @steipete
12. asyncinject 0.7