Dynamic Subagents, Review Loops, and Safer Codex Permissions

Coding Agents Alpha Tracker

2026-06-30

Dynamic Subagents, Review Loops, and Safer Codex Permissions

By Coding Agents Alpha Tracker • June 30, 2026

LangChain's dynamic subagents are the clearest signal today: coding-agent orchestration is moving out of prompt craft and into explicit code. The practical follow-ons are repeated review loops, clean-machine verification, least-privilege permissions, and tighter control of context and token spend.

TOP SIGNAL

Colin at LangChain just productized the biggest shift in coding agents: **orchestration is moving out of prompt prose and into code**. Dynamic subagents in Decode/DeepAgents let the main agent write loops, branching, retries, and parallel fan-out itself [1]. Pair that with Boris Cherny saying the next Claude Code runs subagents in the background by default [2] and Peter Steinberger's production habit of re-running `/review` in fresh sessions until it stops finding new issues [3], and the practical takeaway is clear: the best setups now look like explicit workflows with explicit verification, not one giant autonomous chat.

TRY THIS

- **Write the loop, don't narrate it (Colin, LangChain).** Put the `workflow` keyword in your request so the agent knows to write code and spawn subagents [1]. Build with code interpreter middleware so the `task` global exists—Decode ships with this by default—and have the agent call `await task(description, subagent_type, optional_response_schema)` [1]. Good starter prompts: `run a workflow to review every file in this PR and give me one summary of the problems` for fan-out + synthesize, or `run a workflow`

`to find all the dead code in this repo and don't stop until a full pass turns up nothing new` for exhaustiveness [1].

- **Turn `/review` into a terminating loop (Peter Steinberger, OpenClaw).** Start a fresh session, run `/review`, inspect or fix what it finds, then open another fresh session and run it again; Steinberger says repeated passes keep surfacing new edge cases because coding agents are unpredictable [3]. Then codify that as an `auto review` skill: tell the agent to keep invoking the review CLI in new sessions until it finds nothing new and your invariants pass [3]. Expect 5–10+ iterations on serious changes—he says one session ran 5–10 hours, but it materially raised shipping confidence [3].
- **Make the agent prove it off your machine (Peter Steinberger).** Give it fresh Linux/Windows/macOS boxes so it installs and runs what it built on clean machines, then add browser automation plus computer vision for end-to-end UI checks like verifying that a Slack message actually appears correctly [3]. This is the cleanest way in today's stack to extend verifiability and kill *works on my machine* false positives before merge [3].
- **Trim response and memory overhead with Caveman (Julius Brussee).** Install with `claude plugin marketplace add JuliusBrussee/caveman && claude plugin install caveman@caveman`, or drop `Respond like a caveman. No articles, no filler words, no pleasantries. Short. Direct.` into `CLAUDE.md` [4]. Then run `/caveman-compress path/to/CLAUDE.md` to shrink memory files; the guide says that cuts those input tokens by roughly 46%, while repo benchmarks show 22–87% output compression and 4–15% realistic whole-session savings [4]. For cheap repo spelunking, `cavecrew-investigator` returns read-only `file:line` lookups with about 60% fewer tokens than vanilla Explore [4].

WHAT SHIPPED

- **DeepAgents / Decode — dynamic subagents.** Main agent now writes orchestration code instead of coordinating only through tool calls, which LangChain says enables deterministic coverage across hundreds of documents or thousands of datapoints [5]. The six patterns shown today: classify-and-act, fan-out-and-synthesize, adversarial verification, generate-and-filter, tournament, and loop-until-done [1].
- **Claude Code (next version) — background subagents by default.** Boris Cherny says subagents will run without blocking the main conversation; tell Claude explicitly if you want foreground execution [2].
- **Codex — least-privilege permissions plus usage fixes.** OpenAI replaced coarse sandbox modes with reusable, inheritable permission profiles that bind OS-enforced file rules—including `**/*.env` deny rules—to per-domain network access and Unix sockets, with fail-closed admin al-

lowlists. Docs: developers.openai.com/codex/permissions [6]. Separately, the Codex team says they fixed overeager auto-review/subagent background usage, duplicate suggestions, retry behavior, and misreported turn graphs; `/goal`, subagents, and higher reasoning levels still intentionally consume more capacity [7].

- **Ornith-1.0 — new MIT open-weights coding model.** DeepReinforce released 9B/31B Dense and 35B/397B MoE variants built on Gemma 4 and Qwen 3.5, with claims of state-of-the-art coding performance among comparable open models [8]. Simon Willison says the 35B Q4_K_M GGUF (20GB) worked well with his Pi agent harness across many tool calls and handled codebase-navigation tasks against Datasette with ease. More: simonwillison.net/2026/Jun/29/ornith [8].
- **Mobile agent control is becoming real product surface area.** Cursor for iOS shipped always-on cloud agents, remote control of desktop agents, Live Activities, and phone-side demo/diff review before merging PRs; details: cursor.com/blog/ios-mobile-app [9, 10]. Theo also showed T3 Code + T3 Connect running multiple projects across four computers from one app, with thread-level selection of the exact repo clone/machine; it is fully open source and BYO-subscription across Codex, Claude Code, OpenCode, Grok CLI, and Cursor [11, 12, 13].
- **Caveman — lightweight Claude Code compression plugin.** Julius Brussee's repo packages response compression, `CLAUDE.md` shrinking, and specialized subagents behind a simple Claude Code plugin. Benchmarks in the repo average ~65% output compression, but the guide's more honest number is 4–15% realistic session savings [4].
- **Cost reality check on multi-model subagents.** Cursor builder Jediah Katz says he regularly asks Cursor to `have Opus give a second opinion on our plan` [14], but warns that a cache miss costs about 10 cached steps and subagent telephone overhead can still be worse, so benchmark the split instead of treating cache preservation as doctrine [14].

GO DEEPER

- **3:59–4:39 — Colin on why dynamic subagents matter.** Best short clip for the day's core shift: reliable coverage, branching, retries, and parallelism move from agent reasoning into ordinary code [1].

Dynamic Subagents: How to Run Parallel Agents Reliably in Deep Agents (3:59)

- **7:52–8:48 — Ben Geist at Ramp on shared global context.** If you're building supervisor/worker systems, watch this one: initialize workers from a filtered global KB cache instead of letting each rebuild context from scratch. In his tests, that cut worker tokens 42–57% and total tokens 21–31% at the same accuracy [15].

Intelligence Efficiency, Ben Geist | Compile 26 (7:52)

- **9:41–10:13 — Peter Steinberger on the auto-review loop.** Tiny clip, big idea: keep re-running the review CLI in new sessions until it stops finding novel problems, instead of trusting a single pass [3].

OpenClaw Creator + Snowflake CEO: The Future of AI Software | Snowflake Dev Day 2026 (9:41)

- **20:47–22:44 — Aaron Levie on the *coding harness* backbone.** Strong enterprise frame: start with a sandboxed expert coder that can use tools and connectors, then layer domain expertise on top. That is his template for broader agent design, not just engineering copilots [16].

Why Enterprise AI Adoption Is Slower Than You Think — Aaron Levie (Box) + Harrison Chase (20:47)

- **Repo to study — Caveman.** It is a clean example of using `SKILL.md` plus Claude Code session hooks to change behavior across sessions, not just in one prompt. Start with `/caveman-compress`, `cavecrew-investigator`, and the benchmark folder [4].

Editorial take: the edge is moving from model picking to loop design—who owns the workflow, how context gets passed, and what proves the result before merge. [1, 15, 3]

Sources

1. Dynamic Subagents: How to Run Parallel Agents Reliably in Deep Agents
2. X post by @bcherny
3. OpenClaw Creator + Snowflake CEO: The Future of AI Software | Snowflake Dev Day 2026
4. The Complete Guide to Caveman
5. X post by @LangChain
6. X post by @thsottiaux
7. X post by @thsottiaux
8. Ornith-1.0: Self-Scaffolding LLMs for Agentic Coding
9. X post by @cursor_ai

10. X post by @cursor_ai
11. X post by @theo
12. X post by @theo
13. X post by @theo
14. X post by @jediahkatz
15. Intelligence Efficiency, Ben Geist | Compile 26
16. Why Enterprise AI Adoption Is Slower Than You Think — Aaron Levie (Box) + Harrison Chase