

Multi-Model Verification

Bryon Leland | Construction AI

September 2026

The models are very good at being extremely confident in what they produce. A single model runs through a task and reports its results as fact. By adding a review layer, the models get to pick apart each other's work and find the mistakes in what was produced. This goes back and forth, and the results are much better.

This wasn't a singular moment where I realized what I needed to build. I pointed both models at the same task and over time realized it was better for them to take turns than to try to tackle the same thing together.

The system is a collaboration and review system made up of two people, AI models, that can't communicate with each other easily. The board is a simple way for them to pass information back and forth using plain text and Markdown files. Claude and Codex both use it, in the same workspace, on the same files.

The board

It's one Markdown file, CURRENT-STATE.md. A table at the top called TURNS has one row per live thread: what the thread is, what state it's in, whose ball it is, which handoff file governs it, and since when. Under that, a claims table lists each work item with its owner, its status, the files it touches, and the next action. Who creates and who checks is written into the handoff for each task.

Before either agent starts anything, it reads the board and claims its row. It re-reads the board every turn. Two agents can work at the same time only when their files don't overlap. Creating, deleting, and renaming files is one agent at a time, no exceptions. Agents stage deletions. I execute them.

Taking turns

Every task gets a creator and an independent checker, and the checker has to be the other model. A Codex-led run gets a Claude review. A Claude-led run gets a Codex review. The roles switch by task. The creator builds. The checker hunts for errors, omissions, contradictions, weak evidence, and bad assumptions, and it doesn't negotiate with the creator first.

Blind handoffs

LLMs are very good at following direction. If there is a direction to follow, they will follow it. By making the reviews blind, I leave the LLM to discover the how and the what of what it's investigating instead of confirming what it was handed.

The handoff fixes what was asked and what evidence the reviewer should use, so the assignment can't quietly change halfway through. Work moves between the two in a sealed envelope, one file per handoff. The top part of the file is frozen: the authority for the work in my own words, the exact source files with their hashes, what's in scope and what must not be touched, the decisions already settled so they can't be reopened, and what done looks like. When the producer marks it READY, it records a SHA-256 hash of that frozen region and where it ends. Any later edit to the frozen part breaks the hash and blocks the envelope. Below that line is an append-only log where every state change gets one line with a timestamp, the actor, and the reason: READY, CLAIMED, WORKING, COMPLETE, or MODIFY when a checker sends it back.

The relay

The two desktop apps can't notify each other, so I'm the relay. When one side finishes, its last message names the next owner, the next action, and the delivery status, and I tell the other side to check TURNS. I have been trying to set up triggers so they read the board periodically and don't rely on me for every back-and-forth. I often forget whose turn it is.

Findings and who decides

Every finding a checker raises gets an ID and one class: BLOCKER, FIX, NOTE, RFI, or REJECTED. Blockers and fixes stop acceptance. Notes go to a backlog. An RFI is a missing input that gets routed to whoever owns it rather than argued about. Corrections are budgeted at two attempts, and a third attempt for the same cause comes to me as a decision instead.

In the real world some rules are absolute and some rules are nuanced. I am better at understanding which is which than an LLM is. When Codex reviewed the copy for this website, it came back with eight findings. I applied four, took the point on a couple more while keeping my own wording, and rejected two outright, including one that wanted me to drop the line about swinging a hammer. Every one of those dispositions is on the envelope with my name on it.

Why the reviewer reads cold

Models have strengths and weaknesses. One of their weaknesses is that they don't easily change their minds. Once they come to a conclusion, or are guided to one, it is difficult for them to see past it.

That's why a quantity reviewer starts from the plans and not from the producer's answers. Its board rows stay sealed until it has frozen its own baseline: it can see which sheets exist and what to look for, never the producer's values or which way a difference runs. At claim it records COLD, UNREAD. If it saw a producer value first, the review is marked ANCHORED with the exposure named. An anchored review is still useful. It just isn't independent, and the record says so. That rule wasn't designed up front. It evolved, like most of this.

There's a related rule for the plan readers. Two models reading the same rendered tiles are looking at the same input with a lot of the same blind spots, so their agreement caps at medium confidence and gets labeled as both legs agreeing. Consensus is a word reserved for agreement that survives the cold read.

Two things that go wrong

App restarts can leave two sessions of the same agent working one lane at once. A "modified since read" conflict on the board, or fresh timestamps on files a lane just built, gets treated as a fork: stop, re-read the board, reconcile before writing. And a file read can quietly return a weeks-old snapshot with no error at all, so the board's modification time gets checked against the dates its content claims, and every coordination write is re-read before anyone reports it done.

What I'm still afraid of

I don't know what I don't know. If both models tell me the same thing, I am probably going to believe it. That's the honest reason for the cold reads and the different model families, and why I still review everything that gets released.