Coherence Risk Under Coupling

A controlled Fathom experiment on Cloudflare Workers AI: when an agent's state loses coherence, and how a committed, auditable record restores it.

Embedded Risk Analytics · A Fathom case study · July 2026

Every team building a long-horizon agent has to decide how the agent remembers its own work, and the decision is usually treated as a question of model quality or memory product. The mechanism is different. When the state an agent maintains is coupled, meaning each value it tracks is computed from others, as in a ledger, an evolving plan, or a document full of cross-references, coherence becomes a question of how strongly those values feed one another and whether the record that holds them can keep up. This briefing measures that directly. On a controlled substrate running live on Cloudflare Workers AI, we isolate the conditions under which an agent's running state loses coherence, and we test whether externalizing that state to a committed, periodically re-grounded record restores it.

One result stands out. Past a coupling threshold, the agent's state loses coherence whether it is held in the model's own conversation or in an external store. Coupling drives the failure, and both ways of holding state cross the threshold together. Grounding the record back to truth arrests the failure, and the audit that does so can be sparse: re-grounding as rarely as one turn in ten holds the system well below the point of collapse. The committed, auditable record carries a further advantage, in that it can be inspected and re-grounded on a schedule.

The terms used in this briefing

This is a technical result with a commercial consequence, so we define the vocabulary once, plainly, before using it.

Term	Meaning
Order parameter $\|\lambda\|$	A single number for how persistently a per-turn error carries into later turns. Below about 0.9 the state settles, near 1.0 it sits at criticality where errors hold steady, and above 1.0 it diverges.
Coupling (Λ)	How strongly the agent's state variables feed one another. Here Λ equals fan-in divided by 5, so Λ below 1 is sub-critical and Λ at or above 1 is supercritical.
Reflexive context	The agent keeps state in its own growing conversation and re-reads what it previously wrote.
Committed store	The agent writes each state value to an external durable record and reads it back, an auditable database that stands apart from the conversation.
Re-grounding (cadence k)	Periodically overwriting the committed record with the audited true value, once every k turns.
Feasibility inequality	Coherence holds only while record-layer capacity C covers exogenous uncertainty H_{ext} plus the reflexive burden B_ρ, written $C \geq H_{ext} + B_\rho$.

Why coupled state is where agents lose coherence

A long-horizon agent accumulates state as it works, and over a long run it must keep that state internally consistent. The failure most teams picture is simple forgetting, where a value drops out of the window and is lost. A harder failure runs underneath it. When the values an agent tracks are coupled, a small error in one propagates: it is read back in, folded into the next value, and carried forward. Whether those carried errors fade or compound follows from how tightly the state is coupled relative to the capacity of the record that holds it. This is the regime Embedded Risk Analytics' feasibility inequality describes, where coherence survives only while record-layer capacity covers both the uncertainty arriving from outside and the burden the system places on itself by feeding its own state back in. The experiment below puts numbers on that boundary.

How we isolated the effect under control

To measure the mechanism cleanly we built a deliberately contrived substrate: a ledger of interdependent integer variables, where each turn the agent recomputes one variable from a fan-in of several others under a fixed aggregation rule. The task, the model (Llama-3.3-70B on Cloudflare Workers AI), the horizon of ninety turns, and the way the agent is prompted stay fixed. Two things vary: the coupling, set by how many variables feed each update, and how the agent holds state between turns.

Three state architectures are compared. **Reflexive context** keeps the ledger in the model's own growing conversation. The **committed store** writes each value to an external durable record and reads it back. A **truth-fed control** receives the correct state each turn, which severs any carried error and isolates the burden an architecture adds from the instability the task would show on its own. We read a single order parameter, $|\lambda|$, from each run's per-turn error trajectory, and we report its median across sixteen independent seeds per condition with bootstrap intervals. The substrate is synthetic and the conditions are matched, so the result speaks to the mechanism itself. The elicitation and the per-turn outputs are version-controlled and audited, so the measurement tracks the mechanism and stays clear of prompt effects.

Past a coupling threshold, coherence poles wherever state lives

At low coupling every architecture settles: errors made in one turn wash out by the next, and $|\lambda|$ sits near zero. As coupling rises, that changes. Once the coupling passes $\Lambda \approx 1.2$, both unaudited architectures climb to criticality together. The reflexive context and the committed store both reach $|\lambda|$ near 1.0, and their steady error grows from a fraction of the true value into the hundreds. At $\Lambda = 1.4$ the reflexive context sits at $|\lambda|$ 1.02 and the unaudited store at 1.03, which are statistically indistinguishable. Coupling drives an agent's record incoherent, and both ways of holding state cross the threshold together. Figure 1A and Table 1 show the crossing.

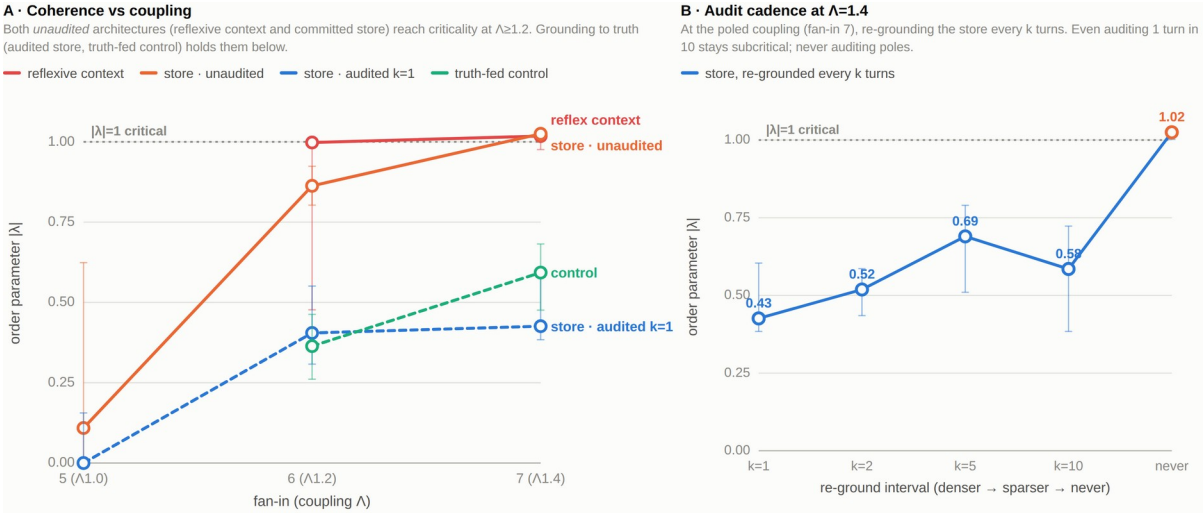

Figure 1. (A) Order parameter $|\lambda|$ versus coupling Λ for four state architectures. The dashed line marks criticality at $|\lambda| = 1$, and both unaudited architectures cross it at $\Lambda \geq 1.2$. (B) At the supercritical coupling $\Lambda = 1.4$, $|\lambda|$ for the committed store as a function of how often it is re-grounded. A sparse audit still holds it below criticality, while never auditing lets it pole. Bars are 95% bootstrap intervals across sixteen seeds.

Table 1. Order parameter $|\lambda|$ by architecture and coupling, with steady error at the poled coupling.

State architecture	$\Lambda = 1.0$	$\Lambda = 1.2$	$\Lambda = 1.4$	Steady error (Λ=1.4)
Reflexive context	n/a	1.00	1.02	287
Committed store, unaudited	0.11	0.86	1.03	230
Committed store, audited (k=1)	0.00	0.41	0.43	47
Truth-fed control	n/a	0.36	0.59	33

Re-grounding the committed record arrests the pole, and the audit can be sparse

The same experiment shows what restores coherence. Grounding the record back to truth pulls the system off the pole. The truth-fed control stays in the settled regime, holding $|\lambda|$ at 0.59 even at $\Lambda = 1.4$ where the unaudited arms diverge. Re-grounding the committed store does the same, and it works without auditing every step. Re-grounding once per turn holds $|\lambda|$ at 0.43, and re-grounding only once every ten turns still holds it at 0.59, against 1.03 with no auditing. The audit interval is a dial that trades cost against coherence, and even its sparse settings keep the record well clear of collapse. Table 2 lays out the cadence.

Table 2. Committed store at $\Lambda = 1.4$: coherence as a function of how often the record is re-grounded to truth.

Re-ground interval	Order parameter $\|\lambda\|$	Steady error
Every turn (k = 1)	0.43	47
Every 2 turns (k = 2)	0.52	50
Every 5 turns (k = 5)	0.69	59
Every 10 turns (k = 10)	0.59	68
Never (unaudited)	1.03	230

There is an asymmetry here that carries more weight than the numbers. The committed store is the one architecture of the two that can be audited at a chosen cadence in the first place. A reflexive conversation's internal state can only be checked against truth by a process that already holds the truth, while the committed record can be checked directly, because it is an explicit, external, inspectable artifact. So the committed, re-groundable record earns a distinct property under coupling: it admits a cost-controlled coherence guarantee.

What this means for teams building coupled agents on Cloudflare

If an agent maintains interdependent state over a long horizon, such as a running ledger, an evolving plan, or a specification whose parts refer to one another, its coherence is set by two things: how tightly the state is coupled and how much capacity the record has. Model capability sits to the side of this. The practical posture that follows is concrete. Hold that state in a committed, external record, and re-ground the record to an audited source of truth on a schedule. The cadence is a tunable cost: audit more often near critical coupling, and less often when the state is loosely coupled. On Cloudflare this maps cleanly onto a Durable Object that holds the committed state and a scheduled re-ground that supplies the audit, which is the harness this experiment ran on.

A deterministic account of agent state via ERA's Fathom harness

What this briefing measures is one instance of the Fathom program: the feasibility inequality made operational as a single order parameter, read on a controlled substrate, on live Cloudflare infrastructure. Stating it this way has a purpose. The instrument that quantified the coupling threshold and the audit cadence here is the same instrument ERA uses to read coherence risk on real agent traces. This experiment is the contrived, fully controlled case, where the mechanism shows in isolation and the remedy is measured directly, and it establishes that the harness, the metric, and the committed-state remedy run end-to-end on Cloudflare.