# AI Is Failing, But Not for the Reason People Think

It’s not getting dumber. It’s running on systems that can’t keep everything in sync once things get complex.



THOMAS ROCHA III

APR 10, 2026


In the first week of April 2026, Anthropic’s Claude went down. TechRadar documented the outage live. Users saw persistent loading indicators, failed responses, and broken voice interactions. DownDetector registered more than 3,000 reports in the first wave. Within 24 hours, it happened again. Business Insider confirmed the scope.

These are not isolated incidents. They are repeated manifestations of the same failure class across independent systems.

Tom's Guide documented the ChatGPT pattern: projects not loading, chat history missing, responses not arriving. 10,000 to 15,000 user reports in minutes. Claude Code, the tool-augmented agentic layer, went down in the same window: not the chat interface, the execution layer.

You could read all of this as ordinary cloud downtime. But the failure signature is too consistent across too many independent systems to read that way.

When Cloudflare exhibited instability in late 2025, Reuters reported that major platforms including ChatGPT were sent into a tailspin. TechCrunch documented the cascade. ABC News covered the downstream AI impact. The models were fine. The infrastructure coordinating them was not.

The failure was not in the intelligence. It was in the coordination layer.

That line is the thesis. Everything below is evidence.

---

**What is actually breaking**

If you have ever had to kill an agent process manually because it would not stop retrying, you have seen this failure mode firsthand.

Developers across LangChain and AutoGen GitHub repositories are documenting it in real time: agents stuck calling the same tool repeatedly, burning tokens with no forward progress, and loops that require manual termination. These are not edge cases. They are open issues with active participation from engineers across organizations.

The pattern is consistent. An agent calls a tool. The tool returns partial state, or fails, or returns correctly but is misinterpreted by the next step. The agent retries. The retry triggers downstream retries. A single action becomes an exponential execution tree. OpenAI billing support threads document the cost shape: unexpected overnight spend, API call volumes that bear no relationship to actual work completed. The system is doing labor. It is not producing value.

Call it what it is: **coordination failure, when individually valid actions produce invalid system outcomes due to the absence of shared authority.**

Microsoft Copilot enterprise complaints reveal the multi-step version. Inconsistent answers across the same document. Workflow tasks that start correctly and lose coherence by the third step. This is not a model making things up inside a single response. It is a system failing to maintain continuity of meaning across its own workflow. It is showing up consistently in enterprise IT forums and practitioner communities to constitute a pattern.

Every additional agent, tool, or workflow step increases coordination surface area without increasing coordination authority. The failure modes above do not plateau. They scale.

---

**The shape of the problem**

These are not model quality failures. The models are not getting dumber. The training data is not corrupted. The alignment is not degrading.

What is failing is architectural.

Current AI systems, including multi-agent frameworks like LangChain and AutoGen, and enterprise deployments like Microsoft Copilot, are

built without a shared authority boundary across interactions. Each agent, each tool call, each step in a workflow operates with its own partial view of state. There is no session-scoped control layer maintaining coherent authority over the interaction as a whole.

When two agents act on shared state, each with its own context, those contexts are not guaranteed to be consistent. The actions are individually valid. The combined result is not. This produces the corrupted outputs and non-deterministic behavior that developers describe as “flaky”: systems that pass tests and fail in production.

When a tool fails and an agent retries, there is no session-level authority arbitrating whether the retry is appropriate, how many have already occurred, or what state the downstream system is in. Each retry is locally reasonable. The aggregate is exponential. This produces the cost spikes and runaway loops that require manual intervention.

When a session extends across multiple steps, context is reconstructed from partial signals at each step rather than drawn from an authoritative source. This produces inter-agent hallucination: not a model fabricating facts, but a system failing to maintain continuity of meaning across its own workflow.

The pattern is not a bug. It is a structural property of systems built without session-level governance.

---

**Why this is different from what people call “AI failure”**

Most public discussion of AI failure focuses on model-level problems: hallucination, bias, misalignment, and harmful outputs. Those are real, and they are receiving real attention.

What is described above is different. It is not a failure of what the model says. It is a failure of what the system does.

You cannot fix coordination failure by improving training. You cannot fix runaway execution loops with a better prompt. You cannot fix state fragmentation across workflow steps by scaling model intelligence. These are not model problems. They are architecture problems. And architecture problems do not improve as systems scale. They compound.

More agents mean more conflicting state. More tool calls mean more retry surface. More workflow steps mean more context fragmentation. The failure modes visible today intensify proportionally with every capability improvement the industry is pursuing. The systems designed to do more are doing more work to produce less reliable results. The agents designed to automate workflows are generating demand for manual intervention. The infrastructure built to reduce cost is generating unexpected cost.

---

**The clarifying insight**

The agent cannot govern itself.

Every current approach to agentic AI governance asks the governed entity to participate in its own governance. The agent decides when to retry. The agent decides when to stop. The agent manages its own context. The agent arbitrates its own authority relative to other agents.

This is not a design choice. It is the absence of a design. When no external session-scoped reference frame exists, the only governance mechanism available is the agent itself. The result is exactly what the LangChain issue threads, AutoGen repositories, and OpenAI status logs are showing: loops, fragmentation, cost blowouts, and non-deterministic behavior under concurrency.

What is missing is an architectural description. A control layer that maintains continuous session identity, governs active interactions independently of transport and application layers, and provides an external authority boundary that the agent participates in but does not control.

The distinction between transport continuity and authority continuity is the core insight. A session can be technically alive, the TCP connection intact, the model responding, the API returning 200, and still be authority-broken. The system is present but ungoverned. This is why AI systems hang rather than fail cleanly. This is why they retry rather than resolve. This is why they produce inconsistent results across steps, while each individual step looks locally correct.

---

**What comes next**

The incidents of April 2026 are not anomalies. They are an early signal of a structural failure class that scales with adoption.

The field is investing heavily in capability: better models, faster inference, larger context windows, more agents. The gap receiving no proportional investment is coordination: how authority is maintained across steps, how state is made authoritative across agents, how sessions are governed as bounded entities with coherent identity from start to finish.

The outages will keep coming. The cost spikes will continue to surprise people. The enterprise complaints about workflow inconsistency will keep accumulating.

Coordination is the gap. Naming it precisely is where the work starts.