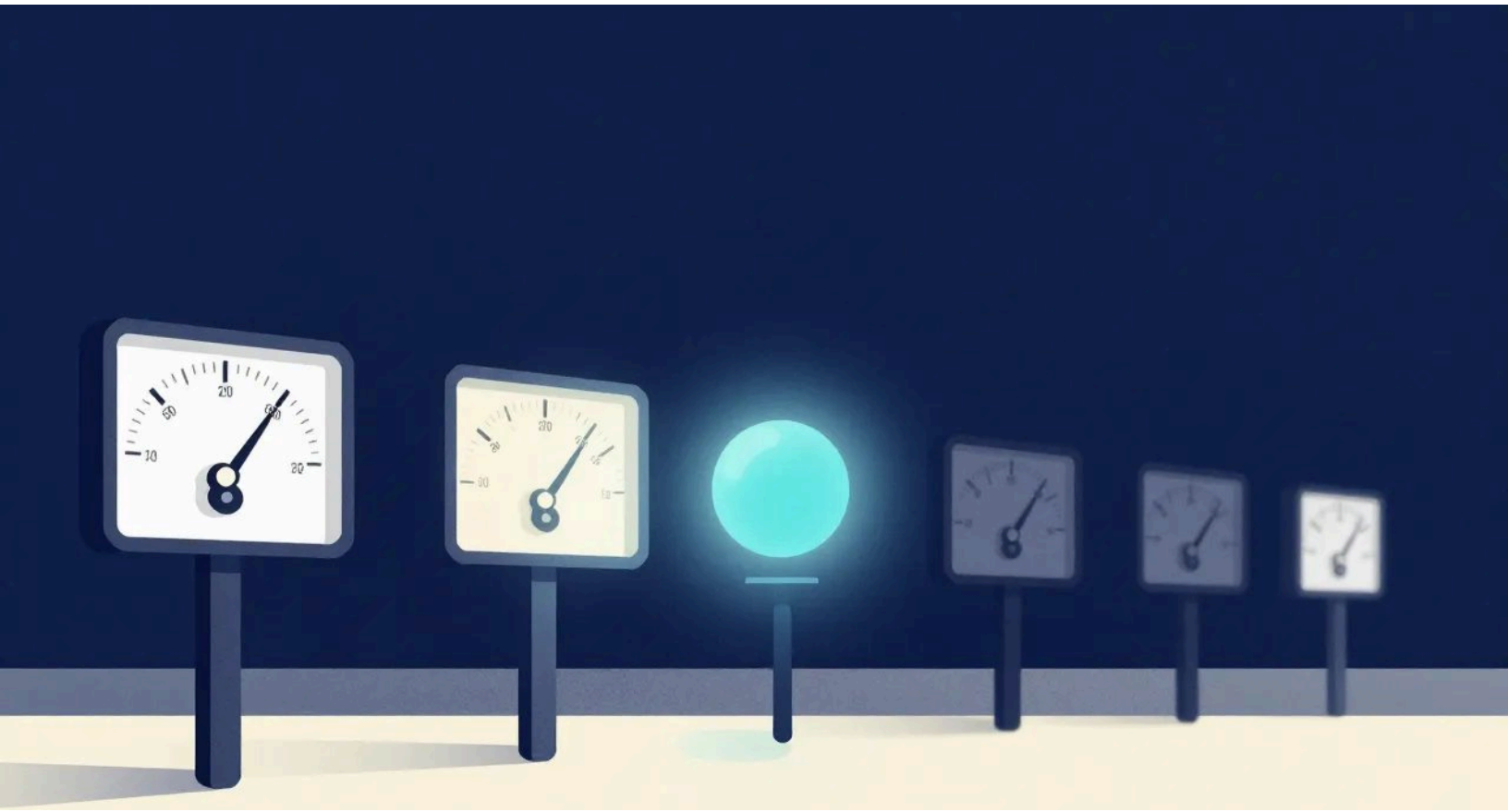


EXECUTIVE PAPER · AI

Evaluations Are the Test Suite for AI

An AI system that has no automated evaluations cannot show it still works after the model, the prompt, or the data changes. CTOs should put evals in the release pipeline with the same authority as unit tests.

AUTHOR

Chris Watkins

SERVICE

AI

FOR

CTO · 13 min read

AT A GLANCE

The problem. AI systems change behavior when anything around them changes: a vendor's model update, an edited prompt, new documents in a retrieval index, a new tool. Most teams find out from customers. In 2025 and 2026 OpenAI and Anthropic both shipped changes that degraded quality for paying users, and both said their evaluations had not been broad or sensitive enough to stop them.

Why it persists. Teams test AI by trying a few prompts and judging whether the answers look good. Those checks are manual, so nobody reruns them when a vendor quietly updates a model. Conventional CI pipelines test code, and most AI regressions involve no code change at all.

The recommendation. Build an evaluation suite from real cases and real failures. Run it automatically on every change to model, prompt, data, or tooling, and block releases that fall below agreed thresholds. Keep scoring a sample of production traffic, and feed every new failure back into the suite. Report pass rates to leadership the way you report uptime.

The Decision in Front of You

Your organization probably runs several AI features already: a support assistant, a document summarizer, an agent that drafts code or answers policy questions. Each one passed some form of review before launch. The decision now is whether you can prove, on any given day, that each one still behaves the way it did when it was approved.

Ordinary software gives you that proof through automated tests. A change that breaks a test does not ship. AI systems need the same discipline, with a difference that matters to the budget: their behavior can change without anyone on your team touching the code. A model vendor retrains, a colleague edits a system prompt, a policy document in the knowledge base gets replaced. Each of these can change answers your customers rely on.

The stakes are legal and financial as well as technical. In February 2024 a Canadian tribunal held Air Canada liable for a refund policy its website chatbot described incorrectly, and rejected the airline's argument that the chatbot was a separate entity responsible for its own statements [6]. In October 2025 Deloitte agreed to repay the final installment of an AU$440,000 government contract after a report prepared partly with a GPT-4o-based tool chain was found to contain fabricated citations and an invented quote from a court judgment [5]. An academic reader found the errors after publication [5].

What's Happening

The model vendors keep finding regressions after release

OpenAI, April 2025. On April 24 and 25, OpenAI shipped an update to GPT-4o in ChatGPT that made the model noticeably more sycophantic, meaning overly eager to agree with and please the user [1]. OpenAI began rolling back to the previous version on April 28 [1]. Its postmortem says the offline evaluations looked good and an A/B test showed users liked the new model. Some expert testers said it felt slightly off, and the company launched anyway [1]. No deployment evaluation tracked sycophancy specifically [1]. OpenAI committed to treat behavior problems such as hallucination and deception as launch-blocking even when A/B metrics look good, and to improve its offline evaluations [1].

Anthropic, August and September 2025. Three separate infrastructure bugs degraded Claude's responses for several weeks [3]. One routed some requests to servers configured for a different context window and affected 16% of Claude Sonnet 4 requests in the worst hour, on August 31 [3]. Another, a server misconfiguration, corrupted output for Claude Opus and Sonnet models during late August and early September [3]. Each bug showed different symptoms on different platforms, so user reports did not point to a single cause [3]. Anthropic said it would run more sensitive evaluations in more places [3].

Anthropic, March and April 2026. Users of Claude Code reported weeks of declining quality before Anthropic published a postmortem on April 23, 2026 [2]. The causes were three product changes: a lower default reasoning setting, a caching bug that discarded the model's working context after an hour of idle time, and a system prompt that capped response length [2]. The model weights had not changed [2]. Anthropic's coding evaluations caught a 3% quality drop from the prompt change; the article does not say how the other two causes were found [2]. Anthropic reset usage limits for all subscribers and committed to broader evaluations for every system prompt change [2].

These companies build the models. Their regressions came from a feedback signal, server configuration, caching, and prompt wording, and none of them required new model weights. A company building on their models carries the same exposure in its own prompts and data, plus whatever the vendor changes underneath it.

Most teams evaluate less than they monitor

LangChain's State of Agent Engineering survey collected 1,340 responses in late 2025 [4]. 57% of respondents had agents in production, and about a third named quality as the main barrier to getting there [4]. Yet only 52% ran offline evaluations against test sets, and 37% evaluated production traffic [4]. Observability was far more common: 89% had some form of it [4]. The sponsor sells evaluation and tracing tools, so treat the figures as indicative.

Tracing records what an agent did. Judging whether that was correct, and whether it was more or less correct than last week, takes an evaluation.

Why the Usual Approach Fails

Most AI features reach production after informal review: a few people try a few prompts and decide the results look right. OpenAI's guidance lists judging quality by impression as an anti-pattern [11]. That review happens once, at launch, and leaves nothing that can be rerun.

Manual review cannot keep pace with change. A system built on a hosted model can see its behavior shift with each vendor release, each prompt edit, and each document added to retrieval. Nobody repeats a manual review for each of those events, so most of them ship untested.

A/B tests and satisfaction scores measure the wrong thing. OpenAI's own users preferred the sycophantic model in testing [1]. People reward answers that agree with them. A quality bar has to measure correctness and policy compliance directly.

Public benchmarks describe the model in general. A leaderboard score says little about whether it applies your refund policy correctly, and OpenAI's guidance also warns against generic metrics, unrepresentative datasets, and evaluating only after release [11].

Standards already expect more. The NIST AI Risk Management Framework, NIST AI 100-1, makes Measure one of its four core functions [7]. Its subcategories call for documented test sets and metrics, for measuring performance under conditions like deployment, for showing the deployed system is valid and reliable, and for monitoring behavior once the system is in production [8]. ISO/IEC 42001:2023, the AI management system standard, requires organizations to decide what to monitor and measure and by what method, to keep evidence of results, and to reassess AI risk when a significant change is proposed [9]. An auditor working from either document will ask for test results tied to each change.

The Framework: The Change Gate

An evaluation, or eval, is an automated test for an AI system: a set of inputs, the expected behavior for each, and a grader that scores the output. Graders can be code that checks an exact answer, a second model that scores against a rubric, or a person [10]. Anthropic's guidance distinguishes capability evals, which measure how well the system handles hard tasks, from regression evals, which should pass close to 100% of the time and exist to catch backsliding [10].

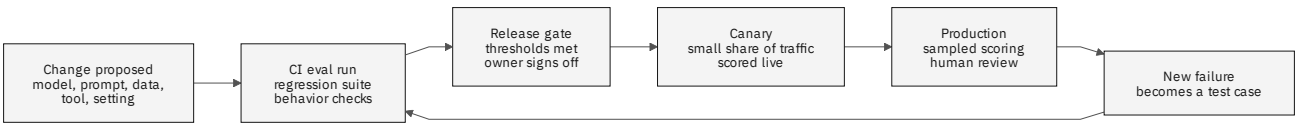
The Change Gate treats every change that can alter behavior the same way, whether your team made it or a vendor did.

Exhibit 1. Most changes that alter AI behavior never touch your code

Change	Who makes it	Example	What should run
Model version	Vendor, or your team on upgrade	A provider retrains or retires the model you call	Full regression and behavior suite before adopting; scheduled rerun against pinned and latest versions
System prompt or instructions	Your team	An edit to shorten answers or add a policy rule	Full regression suite in CI before merge
Retrieval data	Content owners	A policy document is replaced in the knowledge base	Targeted suite for questions that cite the changed content
Tools and agent harness	Your team or vendor	A new API, a changed default setting, a caching change	Agent task suite with tool-call checks
Settings	Your team	Temperature, reasoning effort, context length	Full regression suite

Source: author's framework. Example changes drawn from the OpenAI [1] and Anthropic [2][3] postmortems.

Exhibit 2. The Change Gate: every change runs the suite, and every new failure joins it



Source: author's framework, consistent with NIST AI RMF Measure subcategories [8] and vendor evaluation guidance [10][11].

Build the suite from reality. Start with 20 to 50 cases drawn from actual failures and actual user requests, which is enough to detect the large effects early changes produce [10]. Add the questions that carry legal or financial weight: refunds, eligibility, pricing, safety. Each case needs an expected outcome a reviewer would agree with.

Run it in CI. The suite runs automatically on every pull request that touches a prompt, a model setting, a tool definition, or retrieval configuration. Automating the scoring is what makes it cheap enough to run every time [11].

Gate releases on thresholds. Agree on the pass rate for each suite before the release, and write it down. Regression suites should sit near 100% [10]. Behavior failures such as fabricated facts, policy violations, or unsafe advice block the release outright, which is the rule OpenAI adopted after its rollback [1].

Score production. Run the same graders on a sample of live traffic, and send a share of low-scoring cases to a person for review. Anthropic's guidance pairs automated evals with production monitoring, user feedback, and transcript review, since no single method catches every issue [10].

Calibrate the judges. Model-based graders drift and disagree with people. Check them against human ratings on a fixed sample, and recalibrate when agreement falls [10][11].

A hypothetical case

Consider a mid-size insurer with a customer service assistant that answers coverage questions from policy documents. The vendor ships a new model version, and the team adopts it after a day of informal testing because responses read better.

Three weeks later a complaint shows that the assistant has been telling some customers that water damage from a burst pipe is covered under a policy that excludes it. Nobody can say how many customers heard that answer, or when it started.

With a Change Gate, the model upgrade runs a regression suite that includes the twenty coverage questions most often escalated to agents. The exclusion question fails, the release is blocked, and the team adjusts the retrieval prompt and reruns the suite before switching models. The Air Canada ruling suggests a tribunal would hold the insurer to its assistant's answer [6]. The eval record is how the insurer shows it took reasonable care.

Costs, Risks, and Tradeoffs

Exhibit 3. Evals cost engineering time and slow some releases, and small, low-stakes uses may not need the full gate

Area	Cost or risk	Mitigation
Build effort	Writing cases and expected answers takes domain experts' time, which is scarce	Start with 20 to 50 cases from real failures [10]; grow the suite from production incidents
Run cost	Each run calls the model, and model-graded evals call it again	Run the full suite on release and a smaller smoke suite on every commit
Grader error	Model-based graders can score wrong answers as right	Calibrate against human ratings on a fixed sample each quarter [10][11]
Overfitting	Teams tune prompts to pass the suite and miss new failure types	Keep a held-out set; add fresh production cases monthly
Release friction	Blocked releases frustrate product teams	Agree on thresholds before the work starts; let the owner approve a documented exception
False confidence	A passing suite covers only what it tests, and vendors change things you cannot see [2]	Score live traffic continuously and track complaints against the suite's coverage
When this is wrong	An internal drafting aid whose every output a person reviews and edits, or a short-lived prototype, gains little from a full release gate	Keep a small smoke suite and rely on the human review already in the workflow

Source: author's analysis.

How to Act

Exhibit 4. Every production AI feature has a gated suite within 90 days

Window	Actions	Owner	Milestone
First 30 days	Inventory every AI feature in production, the model and version it calls, and who owns its prompt. Pin model versions where the vendor allows it.	CTO with engineering leads	Inventory complete; every feature has a named owner and a known model version
First 30 days	For the two features with the most customer or financial exposure, write 20 to 50 eval cases from real requests and past failures.	Feature owners with domain experts	Two suites written and passing a first baseline run
First 90 days	Run suites automatically in CI on every prompt, model, tool, or retrieval change. Set written pass thresholds. Make fabricated facts and policy violations release-blocking.	Platform or ML engineering lead	No AI change merges without an eval run on record
First 90 days	Score a daily sample of production traffic with the same graders and route low scores to human review.	Feature owners	Production pass rate reported weekly for each covered feature
First 365 days	Extend suites to every production AI feature. Rerun all suites when a vendor announces a model change. Calibrate model graders against human ratings quarterly.	ML engineering lead with QA	All features gated; grader agreement with humans tracked
First 365 days	Map eval evidence to NIST AI RMF Measure and ISO/IEC 42001 Clause 9. Report pass rates and regressions caught to leadership quarterly.	CTO with risk or compliance	Audit evidence on file; quarterly report delivered

Source: author's recommendation.

Questions to Ask Your Team

1. Which AI features do we run in production, and which model version does each one call today?
2. If our vendor changed that model tonight, how would we know by tomorrow?
3. For our highest-risk AI feature, how many automated test cases exist, and when did they last run?
4. What pass rate does a prompt change need before it ships, and who decided that number?
5. Which customer complaints about AI answers in the last quarter are now test cases?
6. What share of production AI responses do we score, and who looks at the low scores?
7. If a regulator or a court asked how we checked an answer our assistant gave, what would we show?

Closing

No CTO would ship a payment system without automated tests and then let a vendor swap out its libraries without warning. Many AI features run exactly that way today. Treat evaluations as the test suite for AI: build them from real cases, run them on every model, prompt, and data change, block releases that fail, and keep scoring the system in production so the next vendor update is caught by your pipeline before a customer finds it.

Sources

[1] OpenAI. "Expanding on what we missed with sycophancy." 2025. https://openai.com/index/expanding-on-sycophancy/.

[2] VentureBeat. "Mystery solved: Anthropic reveals changes to Claude's harnesses and operating instructions likely caused degradation." Coverage of Anthropic postmortem. 2026. https://venturebeat.com/ai/mystery-solved-anthropic-reveals-changes-to-claudes-harnesses-and-operating-instructions-likely-caused-degradation/.

[3] InfoQ. "Anthropic infrastructure bugs." Coverage of Anthropic, "A postmortem of three recent issues." October 2025. https://www.infoq.com/news/2025/10/anthropic-infrastructure-bugs.

[4] LangChain. "State of Agent Engineering." 2026. https://langchain.com/state-of-agent-engineering.

[5] The Register. "Deloitte refunds Aussie gov after AI fabrications slip into $440K welfare report." October 6, 2025. https://www.theregister.com/2025/10/06/deloitte_ai_report_australia/.

[6] McCarthy Tétrault. "Moffatt v. Air Canada: A Misrepresentation by an AI Chatbot." On Moffatt v. Air Canada, 2024 BCCRT 149. 2024. https://www.mccarthy.ca/en/insights/blogs/techlex/moffatt-v-air-canada-misrepresentation-ai-chatbot.

[7] National Institute of Standards and Technology. "Artificial Intelligence Risk Management Framework (AI RMF 1.0)." NIST AI 100-1. 2023. https://www.nist.gov/node/1731741.

[8] Modulos. "NIST AI RMF: Measure." Summary of NIST AI 100-1 Measure subcategories. https://docs.modulos.ai/frameworks/nist-ai-rmf/measure.

[9] Modulos. "ISO/IEC 42001: Clauses 4-10." Summary of ISO/IEC 42001:2023 requirements. https://docs.modulos.ai/frameworks/iso-42001/clauses-4-10.

[10] Anthropic. "Demystifying evals for AI agents." Engineering at Anthropic. 2026. https://anthropic.com/engineering/demystifying-evals-for-ai-agents.

[11] OpenAI. "Evaluation best practices." OpenAI API documentation. https://developers.openai.com/api/docs/guides/evaluation-best-practices.md.

The views in this paper are the author's own and do not represent any employer. Scenarios are hypothetical. This paper is not legal, financial, or compliance advice.