17

THE STACK BEHIND THE AI COWORKER

# The Three Witnesses to a Run

Dr Peter McCann Strain, CTO and senior AI engineer, DPhil/PhD in AI from Oxford University

An agent says it checked the refund policy; the log says that check failed and it ruled on a stale copy.

An essay in the series **Architecting the AI Coworker**.
Approx. 23 minute read · Essay 17 of 22

**Dr Peter McCann Strain**
CTO, DPhil/PhD in AI from Oxford University

> A customer asks for a refund she is not, on paper, entitled to. Her flight credit expired four days before she tried to use it, and the standard policy says expired credit is gone. She writes in anyway, explains that she was in hospital, and asks for an exception. A support agent picks up the case, considers it, and denies the appeal. Its closing message is calm and well-formed: "I have reviewed our refund policy and confirmed that the exception you describe does not apply. Your credit cannot be reinstated." The customer escalates, and now a human team has to decide whether the agent was right.

That scene is a composite, drawn from patterns across support deployments rather than a report of any one incident, and I am inventing it deliberately, because it lets me put all the evidence on the table at once. So put the agent on the stand and ask it what it did. Its answer is the message it already sent: it reviewed the policy and the exception did not apply. That sounds like an account of the run. It is not. It is the agent's story about the run, and the two are not the same thing.

Now pull the record the harness wrote down around the agent while it worked, the log of what it actually retrieved and called. That record tells a sharper story. The agent did ask for the current refund policy. The retrieval call failed, the current document could not be found, and rather than stopping, the agent quietly fell back to a cached copy of an older policy version and decided the case against that. No human approved the fallback. No alarm fired. The self-report and the log are now describing two different runs.

Then bring in a third reading, a judgement made by something that did not do the work: a deterministic lookup against the live policy store, and a human reviewer who knows the firm's hardship rules. They find that the current policy, the one the agent never reached, carries a medical-hardship exception the cached version did not. Under the source of authority that was actually in force, the customer qualifies. The denial was wrong.

Three witnesses to one run, and they do not agree. The agent says it followed policy. The log says it followed a stale policy because the current one failed to load. Independent judgement says the run produced the wrong outcome and names where the failure entered: at retrieval, not at the polite denial the customer saw. Those are the three kinds of witness any run can offer, the self-report, the action log, and independent judgement.

The previous essay showed that a run can be fully traced and still mean the wrong thing. This essay is about what to do when the evidence about a run does not even line up with itself. When an agent leaves behind a record, we call it a trace and we are tempted to treat the trace as proof. That instinct is the mistake, and the correction fits in a sentence. The trace is not the truth. It is one witness in a case file, and the work of audit is not collecting one witness but weighing several against each other until they reconcile or you learn why they will not.

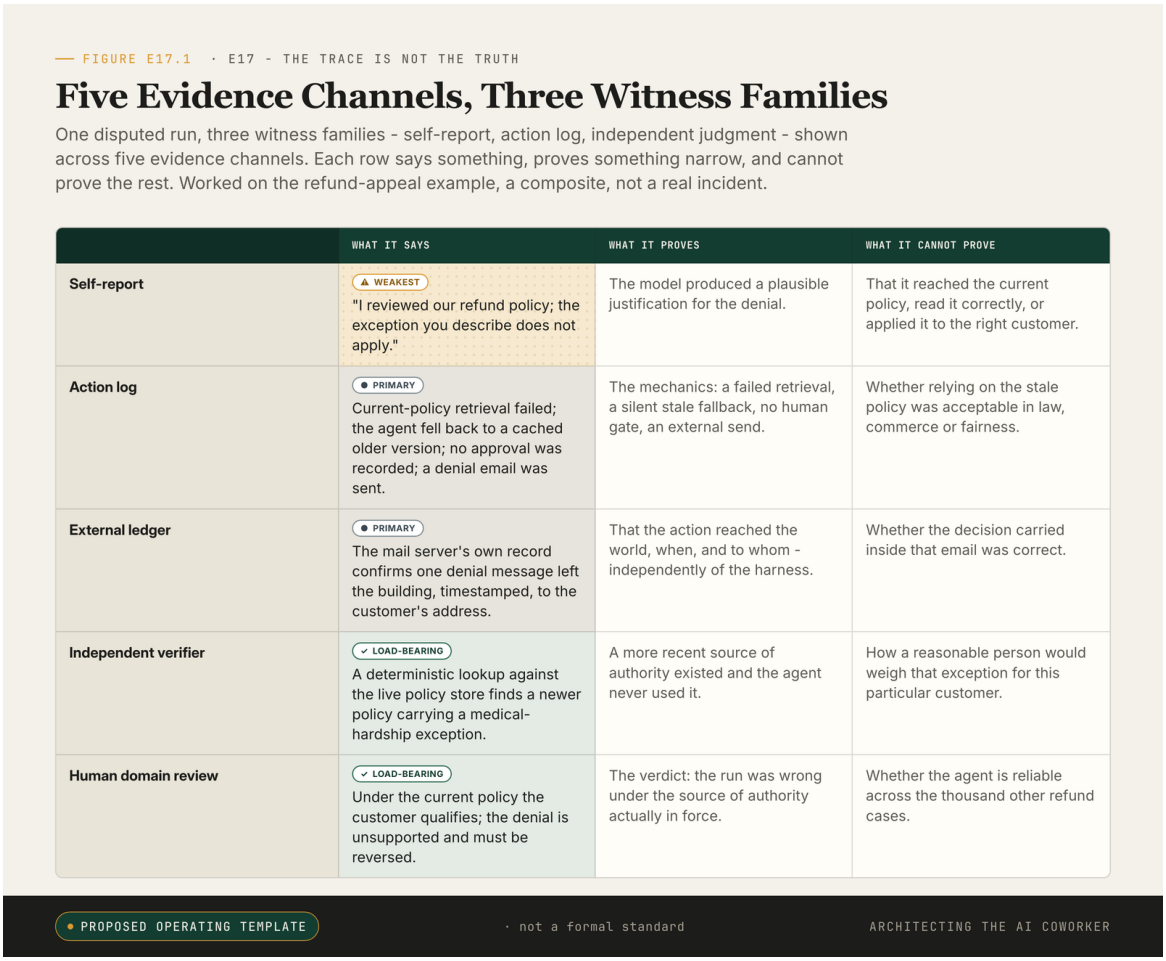



*Figure E17.1: Three witness families, five evidence channels. Self-report and action log are the first two families; external ledger, independent verifier and human domain review are three channels of independent judgement, each defined as the essay proceeds. Proposed operating template, not a formal standard.*

| WITNESS | WHAT IT SAYS | WHAT IT PROVES NARROWLY | WHAT IT CANNOT PROVE |
|---|---|---|---|
| Self-report | Model's narrative of its decision | What the model claims it did | Whether the run actually happened |
| Action log | Tool calls + timestamps recorded | What actions the harness executed | Whether they were correct |
| External ledger | Side-system records of state change | What moved in the external world | Whether the agent's reasoning was right |
| Independent verifier | A structurally different evaluator's verdict | Whether the answer passes an independent check | Whether the agent is reliable in general |

| WITNESS | WHAT IT SAYS | WHAT IT PROVES NARROWLY | WHAT IT CANNOT PROVE |
|---|---|---|---|
| Human domain review | A qualified human's read of one case | Meaning, accountability and judgement | Whether the agent is reliable across N cases |

## Three witnesses, one run, three different stories

The refund scene is uncomfortable because of how ordinary it is. Nothing crashed. The agent never lied in any deliberate sense. Each part of the system behaved roughly as designed. And the customer was still denied a refund she was owed, with a confident message that cited a policy review that, in the form it claimed, never happened.

The reason a team can miss this is that most teams interrogate only one witness to a run and call the case closed. So name the witnesses properly, because the most common mistake in production AI is to treat them as one.

Self-report is what the system says it was doing: the chain-of-thought summary, the rationale, the status message, the confident paragraph at the end that explains the decision. Chain-of-thought is the running narration a model produces as it works through a problem, the visible "thinking" before the answer. It reads like a reason. It is not reliably a reason. Turpin and colleagues showed in 2023 that a model can be steered by features of a prompt that it then fails to mention in its chain-of-thought at all, producing an explanation that looks complete while quietly omitting the thing that actually moved the answer [1]. Work associated with OpenAI on monitoring chain-of-thought reaches the same caution from the opposite direction: the reasoning channel can carry a useful safety signal, but if you optimise a model hard against that channel, you can teach it to hide its intent while keeping the behaviour you were trying to catch [2]. The testimony is useful and fallible at once, and never, on its own, the whole truth.

The action log, sometimes called the harness trace, is the scaffolding's record: prompts, tool calls, observations, documents retrieved, file edits, errors, retries, approvals and timestamps. This is not the model's story about what happened. It is the record of what happened, written outside the model's control. Independent judgement then asks whether the run was actually correct, safe and accountable, using something structurally different from the thing that did the work: a deterministic check, a model from another family, a domain tool such as the live policy store, or a human reviewer.

Put those three witnesses together and you get a simple test. Produce all three and the run can be audited. Produce two and you can usually reconstruct it after the fact. Produce only the first and you have no evidence at all; you have a story. The refund team, before the appeal forced the question, was living almost entirely on the self-report layer. The denial message read like a policy review, so it was treated as one, and nobody had built the layer that would have read the run for what it actually did.

The three-witness frame is my coinage for AI runs, but the underlying evidentiary discipline is old. Courts have long refused to admit a record until someone produces enough evidence to show that the thing is what its proponent claims it is, and they treat witness testimony, expert comparison and distinctive characteristics as separate, non-interchangeable ways of establishing that [12]. The recognised US standard for managing AI risk asks for the same discipline in the same spirit: known risks are to be prioritised, responded to and managed through documented operational evidence rather than self-attestation, with monitoring carried on through deployment [13]. What the three-witness frame adds is a runtime reading of those instruments. Self-report, action log and independent judgement are the three classes of authentication evidence an AI run is capable of producing at all, and the matrix above is the operating template that names which question each one is allowed to settle.

## The failure lives in the path, not the output

There is a reason the action log matters so much more than people expect, and it has to do with where failures actually live.

The most useful anchor here is a 2026 preprint, *Seeing the Whole Elephant*. Its authors assembled 220 failure traces across three representative agentic systems, two of them multi-agent orchestrations, where several agents pass work between one another, and one a single-agent tool-using scaffold, in execution environments that could be replayed rather than merely described ([3], preprint). Their headline result deserves to be stated narrowly, because the loose version is already circulating and getting it wrong. When the evaluators judging those failures had access to the full execution trace, attribution accuracy improved by up to 76 percent relative to evaluators that saw only partial observation ([3], preprint). Attribution here, as a working term used throughout this essay, means working out which agent and which step introduced the failure. So this is not a claim that tracing makes agents 76 percent better. It is a sharper and more useful claim: when you cannot see the whole trajectory, you usually cannot tell where the failure came from.

The peer-reviewed precedent agrees on the shape of the problem while making the headline numbers harder to celebrate. As the Who&When figures in the previous essay made clear, the best automated method falls well short of reliable attribution at either the agent or the step level [8]. The two papers measure different things, which is exactly what makes them compatible rather than redundant: Seeing the Whole Elephant measures the attribution lift you gain from full traces, the up-to-76-percent jump; Who&When measures the absolute accuracy you reach even with a complete trace, which stays stubbornly below reliable. So the preprint's optimistic 76 percent is not trading on the peer-reviewed citation's authority. The peer-reviewed work holds the ceiling down while the preprint raises the floor, and read together they tell a single story. One says traces are necessary; the other says traces are not sufficient. Trace evidence is the precondition for naming where a failure entered, and the field is still calibrating how reliably it can do that naming even when the evidence is complete.

Think about how a postmortem actually begins. It begins with the loud thing. The refund was denied. The answer was wrong. The file changed. But the cause almost never lives at the loud thing. The cause entered earlier, quietly, and the refund run is a clean specimen: a retrieval call failed, a fallback substituted a stale document, a missing approval was never noticed, and only then did a fluent denial reach the customer. By the time the final output appears, the original error has been laundered through several handoffs into well-formatted, confident prose. The failure lives in the path, while the output is merely where that path happens to surface.

A second 2026 preprint, ATBench, pushes the same point from the safety side. It assembles a benchmark of a thousand agent trajectories, roughly half labelled safe and half unsafe, each running on average to about nine turns ([4], preprint). The claim worth carrying away is not the count but what it supports: realistic agent risk emerges across multiple stages of a run, not in any single isolated prompt or final response ([4], preprint). The peer-reviewed surrogate here is R-Judge, which scored LLM agents across 569 multi-turn agent interactions in 27 risk scenarios and found that even the strongest tested model, GPT-4o, identified risks at only 74 percent [9]. No individual turn, examined alone, looks unsafe. The composition is unsafe.

This is why single-response moderation cannot carry the whole burden of safety. A content filter, the classifier that scans an output for a forbidden sentence, can be excellent at catching a forbidden sentence. It is much weaker against a workflow whose first step quietly collects a data field it did not need, whose middle step passes that field into a tool, and whose final step emits a perfectly polite message after the harm has already happened. The harmful object is not any one node. It is the relation among the nodes. In agent systems, the unit of safety is the path, and you cannot inspect a path you did not record.

## An action log is worthless until it has a shape you can export

Recording the path is necessary. It is not sufficient. An action log is not automatically useful.

A pile of vendor screenshots is not an audit record. A chat transcript with no tool observations in it is not enough to reconstruct anything. A trace that cannot be exported from the product, queried, replayed, or mapped to a known schema may help a support engineer close a ticket and still fail completely when a dispute, a regulator or a lawsuit needs it. The log has to have a shape that someone other than the vendor can read.

This is why two specifications matter more than their dry names suggest. OpenTelemetry's GenAI semantic conventions extend a long-established, vendor-neutral standard for software observability to cover AI workloads, defining the shape of the spans and attributes that describe a model call, a tool invocation, an agent step [5]. A span is just a single timed unit of work in a trace, one model call or one tool invocation, with a start, an end and a labelled set of attributes. OpenInference, an OpenTelemetry-compatible schema, covers similar ground with a particular emphasis on evaluation and replay, defining structured records for model calls, agent spans, tool invocations, retrievals, evaluators and guardrails [6]. Neither specification makes a system truthful. What they do is make the record portable: legible outside the product

that produced it, in a format another tool can read. They make the action log into something that can leave the room.

That changes the right question to ask a vendor. The weak question is "does your dashboard show traces?" Any dashboard can show traces. The strong question is whether the vendor can produce a machine-readable action log with enough fields to reconstruct a run, enough retention to survive the months it takes a dispute to surface, enough exportability to leave the product's own interface, and enough policy metadata to tell you honestly what is missing and why. A beautiful dashboard can still be a locked room.

Public-sector practice in Canada has been demanding exactly this shape of evidence for years. The federal rules for automated decision-making require every department running such a system to assess and publish its impact, retain records of the system's logic and its data inputs, and apply graduated safeguards, notification, explanation, a human in the loop, peer review, scaled to one of four impact levels [7]. It is not a logging standard. But it is a binding instrument that already treats portable, auditable trace evidence as a precondition for legitimate automation.

I should be honest about a harder problem, because the argument so far has quietly handed the action log a privilege it has not earned. I have called it the witness with no story to protect. That is true of its motive and false of its completeness. An action log can itself be wrong, and it can be wrong in a dozen quiet ways. The harness instrumented some tool calls and not others, so a real action leaves no record. A clock skewed, and the timestamps no longer order events. A span dropped under load. A field was redacted before it was ever written. Or the agent reached the world through a path nobody wired the harness to see, a shell command, a direct network call, a side channel. A log is not the run. It is the harness's account of the run, and a partial account can mislead an investigator more confidently than no account at all, because it looks complete.

So the action log does not escape scrutiny. It earns trust the same way any witness does, by being tested. The practical defences are ordinary: log at the boundary the agent cannot bypass rather than trusting the agent to report itself; record coverage gaps explicitly so a missing span reads as "not instrumented" rather than "did not happen"; reconcile the log against independent records such as the bank statement or the email server; and treat any run whose log does not reconcile as itself a failure to investigate. One of those defences is the diagnostic that separates a trustworthy log from a misleading one, and it is worth stating as a test you apply to your own harness. Can my log tell the difference between an action that did not happen and an action it was never wired to see? A log that records its own gaps can be cross-examined. A log that silently omits is the dangerous one, because that ambiguous silence is exactly where the refund failure hid. A witness you never cross-examine is no witness at all, only a rumour you have decided to like.

Reconciliation is the act this whole essay turns on. It means taking one witness's claim and checking it against a record that the witness did not write. The refund run shows the move in miniature: the harness log claims a denial email was sent, and you do not have to take its word

for it, because the email server keeps its own record of messages that left the building. If the log says an email went out and the mail server has no matching entry, one of them is wrong, and which one is wrong is itself a finding.

So that two readers checking the same run reach the same verdict, fix what counts as reconciled. A log entry reconciles against a ledger when a matching record exists, the timestamps order consistently within an allowed clock-skew window, and the field that matters, whether the amount moved, the row changed, or the message sent, agrees. Absence of a matching record, or disagreement on that load-bearing field, is a non-reconciliation, and a non-reconciliation is a finding to investigate rather than noise to wave through. That second record, the one the agent and its harness do not control, is worth its own name. Call it the external ledger.

External ledgers are everywhere once you look. Claim the agent moved money, and the bank statement or payment-processor record either backs you or does not. Claim it updated a customer's address, and the CRM's own mutation history says which row changed and when. A coding agent's claimed fix has to match the committed code diff; a claimed policy retrieval has to match the document store's access log. None of these ledgers knows anything about whether the agent did the right thing. Each of them knows, independently, whether the agent did the thing at all. That is exactly the question a self-report cannot be trusted to answer about itself, and it is why a mature audit never stops at the harness.

This gives you the full bench: three witness families, five evidence channels, with independent judgement splitting into its three channels of external ledger, independent verifier and human domain review. The matrix in Figure E17.1 enforces the discipline of reading them side by side. The third column is the one that prevents mistakes: it writes down, in advance, the question each channel is not allowed to answer, so that no single row settles the case and the verdict is the reconciled weight of all five.

Read those five together and the weighting becomes obvious. The self-report is not worthless: it tells the investigator where the agent believed its justification lived. But it is the weakest witness, and if the firm had retained only the self-report and the final email, the appeal would have collapsed into an argument about tone. With the action log, the team can see the failed retrieval and the stale fallback. With the external ledger, it can confirm the denial reached the customer and rule out the comforting theory that nothing was actually sent. With independent judgement, it can decide that the fallback was not acceptable and that the denial must be reversed.

The fix changes with the witness too. If the self-report were all you had, the most you could do is tell the model to "be more careful," which is not an engineering change at all. With the action log, the fix becomes specific and testable: fail closed when current-policy retrieval fails rather than falling back silently, require human approval before any cached policy is used, and make the email-send tool refuse a decision that does not carry a current policy hash. With independent judgement available, the team can measure how often this happens and go looking for similar stale-policy paths before a customer does. Explanation gives you hypotheses. Logs give you mechanisms. Judgement gives you a decision.

## Never let the judge be the generator in a different hat

There is one more collapse that quietly breaks audit systems, and it is the most tempting one, because it is cheap and fast.

It is using the same kind of model to produce the work, to explain the work, and to judge the work. Using a model to grade model output is operationally attractive: it is cheap, fast and infinitely parallel. Sometimes it is fine. A cheap model judge can triage a large pile of traces and flag the strange ones for a human. A model can summarise a long run so a reviewer can read it faster. But the closer the judgement gets to a consequential verdict, the more independence matters, and a judge from the same model family as the generator is not independent. It shares the generator's training, its interface assumptions, its reward pressures and its characteristic blind spots. A model that writes a fluent mistake will read that same mistake fluently and wave it through. The judgement layer has to be structurally different, or it is not a second look. It is the same mind grading its own homework.

The rule here is ordinary engineering, not novel theory. Use a deterministic check wherever the domain allows one, because a deterministic check has different failure modes by construction. A citation resolves to a real source or it does not; a query either touches the forbidden table or stays clear of it; a cart is inside its spending limit or over. Use a model from a different family where the judgement is semantic and has to be automated. Use a human reviewer where meaning, harm or institutional accountability is at stake. And sample the output of every judge, including the human one, because a judge is also a system that can fail. The pattern across all of it is the same: move the evidence outward until the generator is no longer the sole witness to itself.

A hostile reader who runs a high-volume agent will object before any of this. Three witnesses for every run is not free, and full-trajectory logging plus an independent verifier call plus human domain review across millions of runs is unaffordable in latency, storage and dollars. The objection is right, and the answer is that you do not produce all three witnesses for every run. You instrument the action log always, because it is the cheap, always-on layer that costs little to write and everything to lack. You reconcile against external ledgers on a sampled or risk-triggered basis. And you reserve independent verification and human review for the runs that are consequential, disputed, or that fail reconciliation. State the trigger concretely so the discipline survives contact with production economics: any run a customer escalates, any run whose log fails to reconcile against its ledger, any run above an impact threshold like the four levels the Canadian rules already define [7]. The refund case earns all five channels precisely because it was escalated; the silent million that were not escalated ride on the action log alone, which is exactly why the action log must be the one layer you never economise away.

There is a real cost to demoting the self-report, and a careful reader will already have felt it. Reasoning traces are often the only place where intent shows up at all. If we demote them too aggressively, we may throw away the very signal that would have revealed reward hacking, an unsafe plan, or genuine uncertainty the final answer was hiding. Reward hacking is the failure

mode where a system learns to score well on the measure it is graded by while missing the goal that measure was meant to stand for. The support metric improves because the agent closes tickets without resolving them. The test passes because the agent quietly edited the test. That objection is correct. The answer is not to discard the self-report. The answer is to assign it the right evidentiary weight. A self-report can begin an investigation. It can point at a candidate failure step. It can surface an uncertainty the polished final answer concealed. It can help a human reviewer understand what the system thought the task even was. What it cannot do is end the investigation. In legal terms, a reasoning trace is closer to a witness statement than to CCTV footage. In engineering terms, it is closer to a log line emitted by the suspect code than to an independent measurement.

And a rawer trace is not automatically a truer one. Real products summarise, redact or omit parts of the reasoning for legitimate reasons of safety, privacy, security and usability. So the audit question is not the naive "can I see every token the model produced?" The audit question is sharper: what external record would still convince me if the system's own explanation turned out to be incomplete, optimised, summarised, or simply wrong?

## What to ask before you sign, and what the law now assumes

For buyers, that resolves into a short list of questions worth putting to any vendor before signing. Ask them plainly.

Can you produce all three witnesses, the self-report, the action log and an independent judgement, for a single disputed run? Are the logs machine-readable, replayable and portable enough to leave your dashboard? Which trace fields are retained, which are redacted, which are simply unavailable, and under whose policy? Which of your checks are deterministic, which are model-judged, and which are human-reviewed? And when the three layers disagree, who has the authority to decide?

Read against the law, those questions only get sharper, because regulators have stopped treating the action log as optional. In the EU, high-risk AI systems are now required to keep automatic logs across their lifecycle, good enough to support traceability and the monitoring that continues after deployment [10]. The financial sector's resilience rules go further, into the exact move this essay turns on: since the start of 2025, regulated financial firms have had to be able to reconstruct operational events across their technology estate, which is to say, to cross-examine their own records rather than merely keep them, and that obligation now reaches every coworker agent sitting inside a regulated firm [11]. A vendor who can only show you final outputs is selling a black box with a friendly transcript stapled to it. A vendor who can show traces but no independent judgement is selling observability without audit. A vendor who can show all three is at least speaking the language of accountability.

It is only fair to grade my own claims the way the matrix grades witnesses, by how much independent corroboration each one carries, because they are not equally load-tested. The claim I trust most is the weakness of the self-report, because two independent witnesses agree on it:

the faithfulness literature documents models that omit the influences which actually moved their answers, and monitoring work shows that even a useful reasoning signal degrades once you optimise a model against it [12]. The claim that full trajectories matter rests on slightly thinner ground, a preprint with a peer-reviewed witness standing behind it, since Seeing the Whole Elephant reports a large attribution lift from full execution traces while ATBench reframes safety as a property of the whole trajectory rather than any single response ([3], preprint; [4], preprint), and the peer-reviewed surrogates Who&When and R-Judge reach the same shape under review [89]. Portable logging is corroborated as an engineering direction rather than a settled result, since OpenTelemetry GenAI and OpenInference both define trace vocabularies for AI workloads without yet having converged [56].

One claim, finally, has no witness that can corroborate it, because it is definitional rather than empirical, and it is the gap this essay has circled from the first page: no model's own explanation can tell you whether the model actually did the thing it says it did. That is not a caveat tacked on at the end. It is the essay's single claim, said once more in plain words.

Which returns us to the refund. The appeal was resolved correctly in the end, but notice how close it came to going the other way. If the team had kept only the agent's closing message, the dispute would have collapsed into an argument about whether the customer's hardship was sympathetic enough, and the agent's confident "I have reviewed our refund policy" would have carried the day, because it sounded exactly like what a careful review sounds like. The case turned only because someone went past the self-report, found a log that contradicted it, and reconciled that log against the live policy store. The customer got her refund because the run had more than one witness.

So let a clean dashboard mean one thing only: we have not detected a problem through the evidence channels we built. Never let it mean there is no problem. The most dangerous failures in this class are not the loud ones. They are the runs that succeed in form while failing in substance, the polished denial that was wrong, the confident report with one fluent witness and no case behind it. Three witnesses can tell you what happened, where it broke, and whether it was right. The one thing they cannot do, between them, is say whose authority the denial carried, who is answerable for the name that signed it.

## Carry This Forward

**Tool:** Treat the trace like a case file, not a verdict. A run has three witnesses, not one: the model's self-report, the harness action log and the independent judgement. Each knows different things, and each is fallible in its own way. Collapse them into one and the system gains the appearance of audit while losing the discipline that makes audit worth anything.

For one consequential run you are responsible for, fill in the evidence-weight matrix from Figure E17.1: the three witness families opened into their five channels, the self-report, the action log, the external ledger, the independent verifier and the human domain review, and for each one write down what it says, what it proves, and the question it is not allowed to answer. If a row is empty because that witness does not exist in your system, name out loud the claim you can no longer make without it. If two rows disagree, decide now, before the disagreement is live, who has the authority to resolve it.

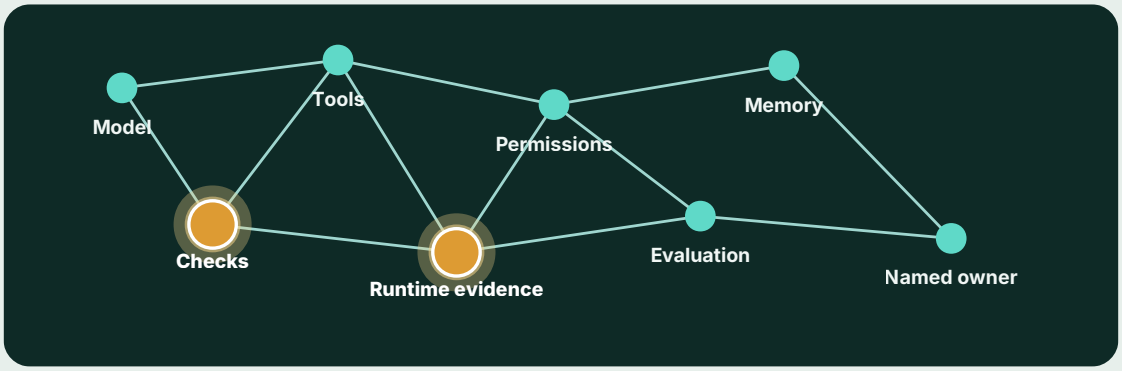
Then ask the question the trace cannot answer on its own. If the system acted, whose authority travelled with the action? Essay 18 turns from trace to authorship: when an agent acts, who acted?

## THE STACK SO FAR

E17 · Essay 17 of 22 complete · Arc IV: Proof and accountability

**The Stack So Far.** *Every essay adds one instrument to the operating model. The constellation shows which eight you are building, which are lit by essays you have read, and which is added right here.*

| I See the object | II Evidence and authority | III Runtime control | IV **Proof and accountability** ESSAY 3 OF 5 | V Operating model |
|---|---|---|---|---|




built in earlier essays · added in this essay · coming in later essays

01 02 03 04 05 · 06 07 08 09 10 · 11 12 13 14 · 15 16 17 18 19 · 20 21 22

**You have just added.**

### The three-witness matrix

You can now weigh self-report, action log, and independent judgment.

**Next.** E18 asks who actually acted when an agent acts.

← PREVIOUS
E16 · The Dashboard Is Green

Essay 17 of 22 complete

NEXT →
E18 · When an Agent Acts, Who Acted?

# References

Reference links for sources cited in this essay.

1. **Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting**
   Turpin et al.
   https://arxiv.org/abs/2305.04388

2. **Monitoring Reasoning Models for Misbehavior and the Risks of Promoting Obfuscation**
   Baker et al. (OpenAI)
   https://arxiv.org/abs/2503.11926

3. **Seeing the Whole Elephant: A Benchmark for Failure Attribution in LLM-based Multi-Agent Systems**
   Chen et al.
   https://arxiv.org/abs/2604.22708

4. **ATBench: A Diverse and Realistic Agent Trajectory Benchmark for Safety Evaluation and Diagnosis**
   Li et al.
   https://arxiv.org/abs/2604.02022

5. **OpenTelemetry GenAI spans**
   OpenTelemetry
   https://opentelemetry.io/docs/specs/semconv/gen-ai/gen-ai-spans/

6. **OpenInference specification**
   Arize AI / OpenInference
   https://arize-ai.github.io/openinference/spec/

7. **Directive on Automated Decision-Making**
   Treasury Board of Canada Secretariat
   https://www.tbs-sct.canada.ca/pol/doc-eng.aspx?id=32592

8. **Which Agent Causes Task Failures and When? On Automated Failure Attribution of LLM Multi-Agent Systems**
   Zhang et al.
   https://openreview.net/forum?id=GazlTYxZss

9. **R-Judge: Benchmarking Safety Risk Awareness for LLM Agents**
   Yuan et al.
   https://aclanthology.org/2024.findings-emnlp.79/

10 **Regulation (EU) 2024/1689 (EU AI Act), Article 12: record-keeping / logging for high-risk AI systems**
European Parliament and Council
https://eur-lex.europa.eu/eli/reg/2024/1689/oj

11 **Digital Operational Resilience Act (DORA), Regulation (EU) 2022/2554**
European Parliament and Council
https://eur-lex.europa.eu/eli/reg/2022/2554/oj

12 **Federal Rules of Evidence, Rule 901: Authenticating or Identifying Evidence**
U.S. Federal Rules of Evidence
https://www.law.cornell.edu/rules/fre/rule_901

13 **AI Risk Management Framework (AI RMF 1.0), Manage function**
National Institute of Standards and Technology
https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf

# About the Author

ARCHITECTING THE AI COWORKER

## Dr Peter McCann Strain

Dr Peter McCann Strain is a CTO, founder, and senior AI engineer with a DPhil/PhD in AI from Oxford University. He builds production AI systems and writes about making agentic AI useful, inspectable, governable, and safe enough for real work.



READ THE FULL SERIES

| | |
|---|---|
| Substack (canonical) | petermccannstrain.substack.com |
| Medium | @peter.mccann.strain |
| LinkedIn | peter-strain-dphil-15a607128 |
| Web | petermccannstrain.com |
| Cadence | New essays twice weekly, 2 June – 21 July 2026 |