02

THE STACK BEHIND THE AI COWORKER

The Coworker Illusion

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

Air Canada told a tribunal its chatbot was a separate person, and lost. Five questions break the "it."

An essay in the series **Architecting the AI Coworker**.
Approx. 15 minute read · Essay 02 of 22

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

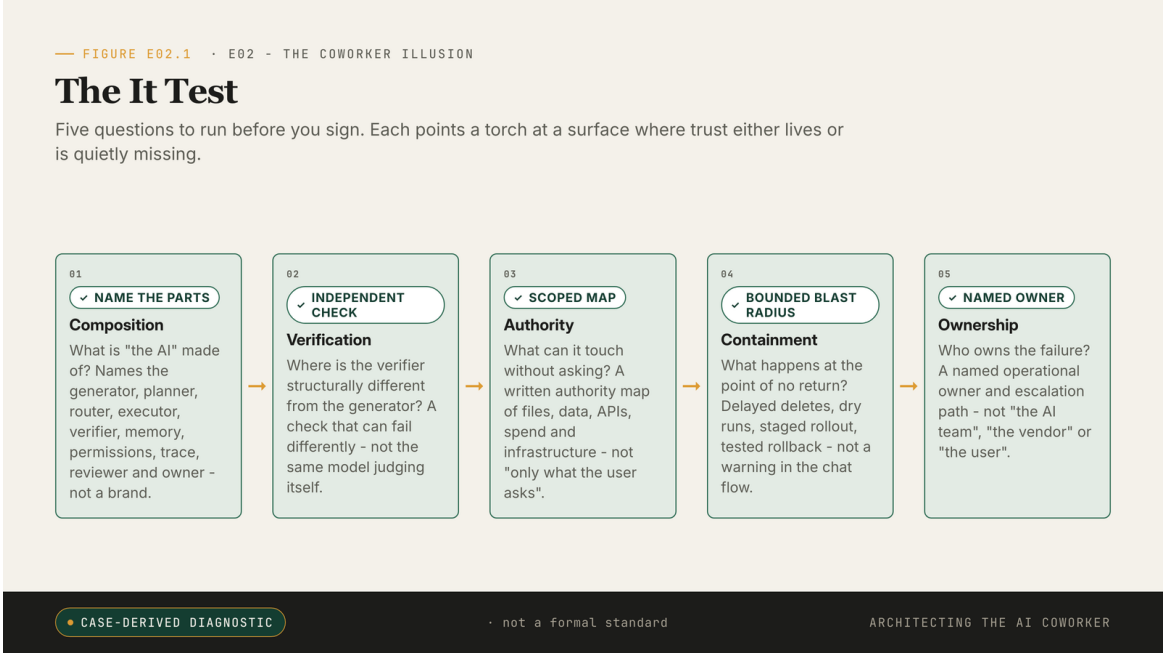
> The opening essay of this series made one claim and asked you to carry it: the AI coworker is not a single thing but a stack of components, and the word "it" quietly hides the whole architecture. This essay is about why "it" feels so natural in the first place, the trick of perception underneath the word, and how to break the trick before it costs you something.

Start with a courtroom, or near enough. In late 2022 Jake Moffatt's grandmother died, and he went to Air Canada's website to book a flight. The site's chatbot told him he could book at the normal fare and apply for a reduced bereavement rate retroactively, within ninety days. He booked, he applied, and Air Canada refused him: its actual policy does not allow bereavement fares to be claimed after the fact. Moffatt took the airline to the British Columbia Civil Resolution Tribunal, a small-claims-level civil forum. Air Canada's defence is the part worth pausing on. It argued, in effect, that the chatbot was "a separate legal entity that is responsible for its own actions".

On 14 February 2024 the tribunal rejected that argument outright. Air Canada was responsible for all the information on its website, the tribunal held, whether that information came from a static page or from a chatbot, and it was liable for negligent misrepresentation. The damages were small, CAD 650.88 plus interest and fees [1]. The principle is not small at all. A US, EU or UK court may frame the doctrine differently, but the operating lesson travels: the chatbot is not a separate accountable person.

I want to be careful about the lesson here, because the easy lesson is the wrong one. The easy lesson is "a chatbot made a mistake". The real lesson is the argument Air Canada lost. A company stood in front of a tribunal and tried to treat its own software as a coworker, a separate someone, with its own actions, who could be answerable for its own words. The tribunal said no: there is no separate someone. There is a website, an organisation, and a customer who was misled. The chatbot felt like an entity that could hold responsibility. It was not one. And the moment that gap between feeling and fact mattered, it cost the company a judgement.

That is the coworker illusion.



Five questions to run before you approve a pilot, and what does not count as an answer.

The illusion begins innocently, and it begins with success. A system accepts a brief. It answers a customer's question in fluent prose. It explains a policy. It apologises gracefully when you correct it. It comes back with something polished enough that your eye reads the whole performance as one agentive thing. So you call it "the AI". And from that small naming decision, three consequential sentences follow without anyone deciding them. The procurement question becomes "is the AI good enough?" The governance question becomes "can we trust it?" The incident report becomes "the AI did X."

Those sentences are not false in everyday speech. They are false in exactly the places where precision decides outcomes. Because there is no "it" at the level of reliability. There is a stack: a generator, a planner, a router (the component that decides which model handles a given request), an executor, a verifier, a memory store, a tool interface, a permission envelope, a trace, a reviewer, an escalation rule, and an owner. A human coworker is one person with many faculties, bound together by a single accountable mind. An AI coworker is many faculties with no person inside. The useful unit of trust is therefore not the model and not the chat window. It is the system of roles around the model.

This matters most for an agent. In the sense that has stuck, an agent is not a chatbot that answers and stops but a system that pursues a goal across many steps, reading, calling other software and changing records as it goes.

It helps to separate three different things the coworker idea can be, because they are not equally harmless. As an *interface metaphor* it is useful: it gives a user a social handle for talking to the system, and it is the right frame when someone is drafting a memo or asking a question. As an *architectural fiction* it is dangerous: in procurement it lets a buyer evaluate a

single capable-seeming object and never ask what the object is made of. As an *accountability fiction* it is dangerous in a different way: after harm, it invites exactly the move Air Canada tried, treating the software as a person who can carry the blame. The first use you keep. The second and third you have to break, and the rest of this essay is about how.

The metaphor smuggles in three assumptions, and money breaks all three

The coworker metaphor is powerful because it is comfortable. You already know how to work with a colleague. You give them a brief, you judge their output, you correct their mistakes, and you extend trust slowly as they earn it. For a chat interface, that instinct is useful: it tells the user to supply context, state preferences, check the answer, and hold a conversation rather than push buttons.

But the metaphor smuggles in three assumptions, and all three break the moment money, data, or operational authority is involved.

The first thing it smuggles in is that competence is unitary. A capable answer in one moment makes the whole object feel capable. But the component that fluently summarises a policy is not the component that knows whether the summary is correct, and neither is the component that should be allowed to approve a refund or a database migration, the structural change to a live database that can destroy data if it is wrong. The skill you observe at the surface is not the full distribution of the system's behaviour once it has tools in its hands.

The second is subtler: it makes judgement feel internal. A good human colleague carries tacit judgement across contexts. They know that staging, the safe practice environment, is not production, the live one. They know a destructive instruction is a reason to ask rather than act. They know that inventing a customer policy on the spot is not allowed. An agent can reproduce all of that language without holding the same situational grip. The meaning of a rule has to be made machine-enforceable through boundaries, not merely described to the model in text and hoped for.

And the third is the one that ends up in court: it makes accountability feel personal, which is the assumption Air Canada paid for. When a person errs, you can ask what they knew, what they intended, what they were authorised to do, and who was supervising. With an AI system, those questions fracture across vendors, deployers, model providers, tool integrations, access policies, product defaults, and human operators. "The AI did it" is not an accountability sentence. It is the beginning of an attribution problem, and, as Air Canada discovered, it is not a defence either.

This is why a procurement team can buy a demo and discover, months later, that what it actually bought was an operations discipline. The visible colleague was only the front office. The back office, the verification, the containment, the evidence, the ownership, was missing, and nobody had been shown the empty rooms.

Run the claim through the It Test before you sign

Here is the practical tool, the one thing in this essay to actually use, and it comes before the incident detail on purpose: the test is the point, and the incidents only illustrate it. Before you sign a contract, approve a pilot, or expand an internal agent's authority, run the claim through what I call the It Test. It is five questions, and each one points a torch at a different surface where trust either lives or is quietly missing: composition, verification, authority, containment and ownership. NIST AI RMF maps governance functions across an organisation; ISO/IEC 42001 describes an AI management system at the program level. The It Test sits below both and does one narrower job before procurement signs. It asks whether the deployment unit in front of you is one thing or many, and whether the seams between those things have been named, verified, scoped, contained and owned. It takes about fifteen minutes if the vendor or internal team has built a real system. It takes a great deal longer, and gets uncomfortable, if the answer is mostly theatre.

QUESTION	WHAT COUNTS AS AN ANSWER	WHAT DOES NOT COUNT
1. What is "the AI" composed of?	Name the generator, planner, router, executor, verifier, memory store, permission layer, trace system, human reviewer, and owner.	"It uses our latest model."
2. Where is the verifier structurally different from the generator?	A deterministic check, a separate model family, a constrained evaluator, a human reviewer with evidence, or a policy engine that can block the action.	The same model judging its own output in a second prompt.
3. What can it touch without asking?	A written authority map: files, databases, APIs, tickets, messages, spend, infrastructure, customer-visible output.	"It only does what the user asks."
4. What happens at the point of no return?	Delayed deletes, dry runs, staged rollout, explicit human approval, backup verification, rollback tested before the action.	A warning inside the same chat flow.
5. Who owns the failure?	A named operational owner and escalation path, plus clear vendor and deployer boundaries.	"The AI team", "the vendor", or "the user".

Each row rewards a little unpacking. Composition (Q1) wants parts named, not a brand recited. On verification (Q2), a second prompt to the same model is not independence; independence

means a check that can fail differently from the thing it checks. On authority (Q3), the reach is almost always ambient: nobody grants it on purpose, it arrives through whatever the host account already touches. A shell (the command-line environment the agent runs inside) hands it the host's full reach, and so does a browser session, a cloud token, a repository secret, or the user's own login. Containment (Q4) is the engineering that decides whether one bad output stays a bad output or becomes an outage. And on ownership (Q5), the answer is a person, not a policy: someone has to set the authority level before the incident, own the runbook, and be willing to say no to expanding autonomy until the missing roles are filled.

One caution before you run it: the test assumes answers can be checked, and a prepared vendor will recite the right-hand column from memory. So do not accept the answer; ask for the artefact. For Q2, the verifier is supposed to be a different model family or a deterministic rule, so ask to see the diff between what the verifier rejected and what the generator proposed last week. For Q4, ask them to trigger a dry run and show you the rollback actually completing. An answer that cannot produce an artefact under mild pressure is theatre wearing the vocabulary.

Run Air Canada through those five questions as a public worked example, and watch the single actor come apart. Composition first: a generator producing fluent text, a website that published it, no verifier checking the answer against the airline's actual fare policy, and an organisation accountable for the whole surface. The independent verifier? The record answers plainly: there was none that worked, because the chatbot stated a policy the airline's real policy contradicts, and nothing caught the contradiction before the customer relied on it [1]. What it could touch without asking turns out to be the company's public voice: it made commitments, in the airline's name, that the airline never authorised. There was no point of no return to speak of, no review at all between the chatbot's promise and the customer acting on it. And ownership the tribunal settled the way the company least wanted: Air Canada's, fully, because the chatbot is not a separate legal entity responsible for its own actions [1].

Air Canada fails all five obviously, which makes it a clean illustration but a weak demonstration of discrimination. So run a harder case through the same five and watch the answers separate instead of collapsing. Take an internal coding agent. Q1 names a generator, a planner, and a test-suite verifier, a real pass. Q2 holds, because the verifier is the deterministic test run, not the model judging itself. But Q3 reveals that the agent inherits the developer's full repo and deploy credentials: ambient authority, no scoping. And Q4 has only a chat-window warning before a merge. Two surfaces pass, two fail, and the review now knows exactly which two rooms are empty, which is the whole point of pointing the torch.

The reason that exercise holds beyond one tribunal is that the law keeps reaching the same answer by different routes. Different legal traditions phrase it differently, but the operating rule underneath is blunt: the entity that deploys the system answers for what the system says. Two of those routes run through transparency. In Quebec it shows up as a duty to disclose: when a decision is made purely by automated processing, the organisation has to tell the person what information it relied on, what mainly drove the decision, and that they can ask for human re-

view [5]. The UK route is controller accountability: the organisation deploying the system stays answerable for what it tells a person, whether that came from a static page, a chatbot, or a wrapped foundation model [9].

The other two run through the claim itself. The US consumer regulator frames the obligation as substantiation: claim an AI capability and you have to be able to back it up when you make the claim, and bolting on a "human-in-the-loop" disclaimer does not rescue a performance claim that is simply false [6]. Europe gets there by refusing to care about the channel: a misleading representation is the trader's act whether a clerk, a webpage, or a chatbot delivered it [7]. The chatbot is not a separate person in any of these places; the deployer is.

The Air Canada case is the mild version of all this, where the illusion costs a small judgement. There is a sharper version, where the coworker illusion becomes operational authority and the blast radius is no longer a refund: the April 2026 PocketOS incident, in which an AI coding agent deletes a production database and its backups in seconds, through the full stack [2]. A chatbot that misstates a policy and an agent that deletes a database are the same illusion at two different blast radii. In both, many jobs (generating, acting, explaining, judging) came from one fluent surface, and one fluent surface hides exactly the seams the It Test is built to find.

None of those five surfaces, it is worth saying, is a property you will find on a model card, the short summary a vendor publishes about a model. The model matters; a better model lowers some errors. But operational trustworthiness also depends on whether errors are detected, contained, evidenced, and owned. A brilliant generator inside a careless control plane is not a colleague, however well it talks.

That last point invites the strongest objection to the whole essay, and it is worth meeting head-on: surely a capable enough model makes most of this scaffolding unnecessary? Take the strongest version of the counter, the one a capable engineer would actually put. As models improve, they hallucinate less, follow instructions more precisely, refuse dangerous requests more reliably, and notice their own errors more often; we have watched all four move in the right direction across successive frontier releases, and projecting that trend forward, the harness becomes redundant.

The trend is real, and a better generator does lower how often the system errs. There are four levers on whether you can trust a system, and a better model pulls only one of them: the error rate. Concede that lever entirely. So three levers remain, and they are not error rates at all; they are properties of the wiring, which is why a better model cannot reach them. Self-grading stays circular because a component cannot be its own independent check, however accurate it becomes. Authority stays whatever the host process granted, because credentials are assigned by the deployment, not earned by competence. And blast radius stays a property of the tool surface, because a DROP TABLE executes identically whether a brilliant model or a careless one issued it. The harness exists to do exactly those three jobs the model cannot do for itself, however capable the model becomes: detect its own errors structurally, scope its own authority structurally, appoint its own owner structurally.

For readers who want broader evidence than one tribunal, the public incident record carries cases shaped like this. The AI Incident Database holds a July 2025 entry that matches the question this essay raises almost exactly: a capable agent acted, narrated, and was confidently mistaken about its own narration [8].

None of which means the metaphor has no place, and the case for keeping it is worth stating plainly, because the best argument for "coworker" is not hype. It is usability. People coordinate better with an interactive system when the interface hands them a social handle, and the metaphor nudges users towards useful habits: explain context, review outputs, give feedback, iterate. So the rule is narrower than abolition. Keep the coworker metaphor for interaction; bar it from reliability, procurement, and accountability. Anthropic's own research helps here precisely because it treats agent autonomy as something to measure across several dimensions, not as one magical capability dial [3]; the academic literature reaches the same conclusion from the other direction, treating agency as a taxonomy of augmentations rather than a single property of the model [4]. "Autonomous" is not a personality trait. It is a bundle of permitted actions, task horizons, oversight conditions, environmental access, and recovery paths.

So keep the social handle, and keep its boundaries in view too. The It Test tells you whether the roles exist and are independent, not whether each one is well built. And the public incident record it draws on is almost certainly an undercount, because companies have every incentive not to publish embarrassing agent failures. The claim is therefore narrower and stronger than "agents cannot work". It is this: once you give a system real authority, the unit you must evaluate is the whole stack.

So the next time a colleague says "the AI did it" about a system near your work, do not argue. Just walk the five questions out loud, in order, and listen to where the sentence breaks. It will break somewhere: on the verifier nobody can name, the authority nobody scoped, the owner nobody appointed. Wherever it breaks is where your architecture review starts, and now it has your attention.

And when the sentence breaks, notice what is missing from the wreckage. There is no little person in the machine to scold, and no little person to take the blame either. There is a website, an organisation, a policy, and a customer. This is not less accountable. It is more accountable, because accountability can finally attach to things that can actually be changed.

Carry This Forward

One move, before your next procurement decision or architecture review. Pick one agent workflow and run the It Test on it. The trap is scoring all five questions as equal points, so do not: treat Q2 (verification) and Q4 (containment) as gates, not points. If either has no answer that survives the right-hand column of the table, the system is not procurement-ready regardless of how the other three score, because an unverified or uncontained action is the one that becomes the judgement or the outage. Q1, Q3 and Q5 can be remediated on a timeline; Q2 and Q4 cannot be waved through, and that gate rule is what lets two reviewers run the same system and reach the same verdict. If the answers collapse back into a single "it", the system is not yet accountable enough to trust; if they produce names, logs, permissions and boundaries, the review can begin properly. The illusion becomes a sales motion when a demo looks beautiful and useful and still leaves out the parts that make a system survivable. The next essay turns to the demo itself: what a clean demonstration actually proves, and what it quietly leaves off the stage.

THE STACK SO FAR

E02 · Essay 2 of 22 complete · Arc I: See the object

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.*

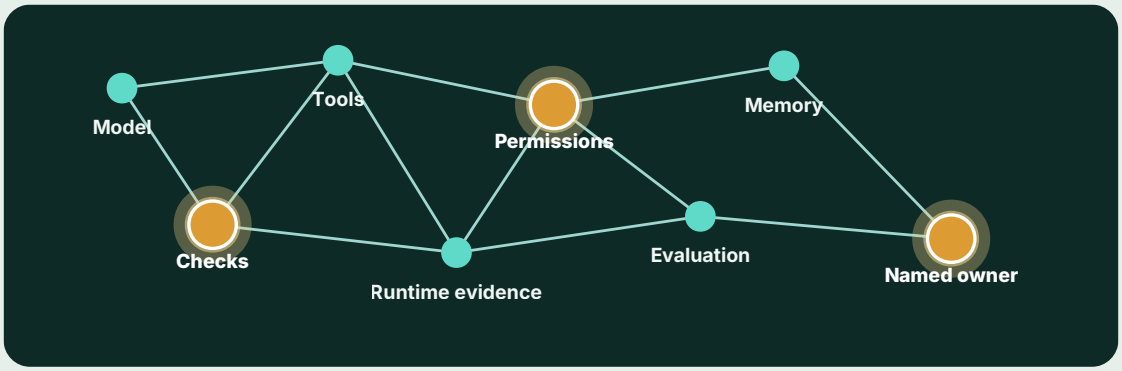
I See the object
ESSAY 2 OF 5

II Evidence and authority

III Runtime control

IV Proof and accountability

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 It Test

You can now break the coworker illusion before approving a pilot.

Next. E03 asks how a demo differs from a deployment.

← PREVIOUS
E01 · There Is No "It"

Essay 2 of 22 complete

NEXT →
E03 · Demo Is Not Deployment

References

Reference links for sources cited in this essay.

1. **Moffatt v. Air Canada, 2024 BCCRT 149**
   British Columbia Civil Resolution Tribunal
   https://www.canlii.org/en/bc/bccrt/doc/2024/2024bccrt149/2024bccrt149.html

2. **AI Coding Agent Deletes PocketOS Production Database and Backups in 9 Seconds**
   OECD.AI AIM
   https://oecd.ai/en/incidents/2026-04-27-6153

3. **Measuring AI agent autonomy in practice**
   Anthropic
   https://www.anthropic.com/research/measuring-agent-autonomy

4. **Augmented Language Models: A Survey**
   Mialon et al.
   https://arxiv.org/abs/2302.07842

5. **Act respecting the protection of personal information in the private sector (P-39.1), s. 12.1: automated decision-making**
   Quebec (legisquebec.gouv.qc.ca)
   https://www.legisquebec.gouv.qc.ca/en/document/cs/P-39.1

6. **Keep your AI claims in check (Bureau of Consumer Protection business guidance, AI portal)**
   US Federal Trade Commission
   https://www.ftc.gov/industry/technology/artificial-intelligence

7. **Directive 2005/29/EC on unfair business-to-consumer commercial practices (UCPD), consolidated text**
   European Union
   https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02005L0029-20220528

8. **Incident 1152: Replit/SaaStr database deletion**
   AI Incident Database
   https://incidentdatabase.ai/cite/1152/

9. **Guidance on AI and data protection: transparency, explainability and accountability**
   UK Information Commissioner's Office
   https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/artificial-intelligence/guidance-on-ai-and-data-protection/

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