22

THE STACK BEHIND THE AI COWORKER

Stop Delegating. Start Architecting.

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

Five questions every Tuesday morning that turn "the AI did it" into a named owner.

An essay in the series **Architecting the AI Coworker**.
Approx. 24 minute read · Essay 22 of 22

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

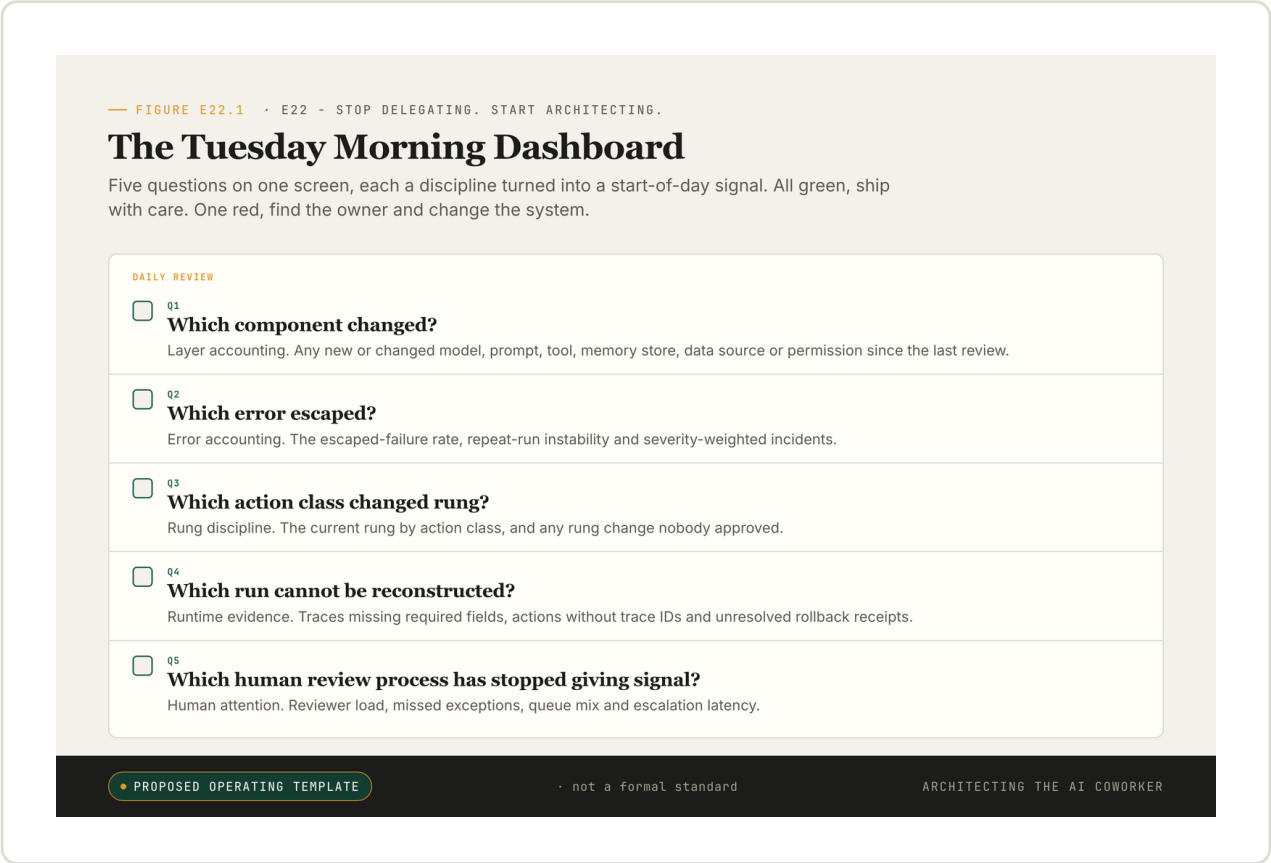
> Picture a Tuesday morning in a company that has put agents into production. The dashboard is green. Nobody looks at it for long, because the first useful meeting of the day is not about the dashboard. It is about one strange run from overnight: the kind of action it touched, the evidence it produced, and the single question that decides how the morning goes. Who can say what happens next?

If a name comes back quickly, the company has a system. If the name does not come back, or three people each assume it is someone else, the company has something that looks like a system and is not one. That difference is the whole subject of this essay, and it is where the series has been heading all along.

This is the twenty-second and final essay of the series, so a reader arriving here cold needs a few terms carried in. An agent, through this series, has meant not a chatbot that answers and stops but a system that takes a goal, plans steps and acts, calling tools, changing records, moving money. A stack has meant the layered thing an agent is, a model wrapped in prompts, tools, memory, permissions and oversight, rather than a single clever object.

Two more terms matter, and each deserves its own breath. Expected harm has meant the figure that actually measures reliability. It is not how often the model is right; it is how often the system errs, how often that error is missed, how often a missed error escapes its containment, and how severe the result is when it does. Hold onto that four-part chain, because the whole operating model turns on it. And the autonomy ladder, from the previous essay, has meant the graded scale of what an action is permitted to do on its own, from drafting only, up through human-approved actions, to bounded autonomy and beyond. That essay ended on a question it could not answer itself: who owns the ladder?

This essay answers it, and the answer is a change of verb. Delegating is what you do with a colleague you have decided to trust: you hand over the work and look away. Architecting is what an agentic system actually requires: you design the team, name its parts, and stay responsible for how they fit. You are not delegating to a model. You are architecting a team.



Five questions on one screen. All green: ship with care. One red: find the owner, change the system.

The model still matters. It was simply never the whole object. The agent does its work inside a system, and it is the system that earns or loses trustworthiness. So human responsibility moves upward, off the keyboard and into architecture, evidence, governance, operation and stewardship. The operating model starts with a humbler act than most AI strategy work ever attempts: name the parts of the system, and name the people responsible for them.

Five disciplines separate a team from a costume

Five disciplines separate the teams I have watched ship reliable agentic systems from the teams that have not. None of the five is a property of the model. All five are properties of the management around it. To avoid a confusion that this kind of writing invites, I will reserve the word *discipline* for these five management practices, and the word *component* for the technical parts of the stack a deployed agent is built from. The disciplines are what people do; the components are what the system is made of.

Layer accounting comes first: keeping an honest map of the components and who owns each. The weakest architecture sentence in AI is "we ship Claude," or "we use GPT," or "we built an agent." It sounds concrete. It hides almost everything that matters. A deployed agent is a stack of components: the model, the prompts, the tools, the retrieval, the memory, the permissions, the orchestration, the evaluation, the telemetry, the user interface, the escalation path, the policy, and the named owner. If a team cannot name those components, it cannot inspect its

failures, and a team that cannot inspect its failures can only improve the system by superstition.

The MIT AI Agent Index is useful here precisely because it documents not only agents but the missing public information about them: it covers around thirty deployed agents and describes uneven transparency around design, safety, evaluation and third-party testing [1]. The exact counts are dated, because the underlying index is still active, but the pattern is enough: low documentation is not an edge case any more, it is becoming the default. The person who holds this discipline is the architect, and the architect is not someone who worships a diagram. The architect is the person who can say, of a given action, "it passed through these components, under these permissions, with these fallback paths, and here is the evidence." The failure mode is single-entity thinking: the organisation says "the AI did it" because it never named the system that made the act possible.

Error accounting is the second. Accuracy is a small number pretending to be a large answer. The reliability question is not "how often is the model right?" It is the chain of four questions named earlier: how often does the system err, how often is that error missed, how often does a missed error escape containment, and how severe is the harm when it does escape? A team that cannot name those four numbers is not managing reliability; it is admiring performance. This is where automation gets cruel. Endsley's retrospective names it directly as the "automation conundrum": as autonomy and reliability rise, operator situation awareness drops and takeover capability erodes, so the human's remaining job becomes rarer, less practised and more consequential [2]; the older systems vocabulary makes the same point another way, that what matters is not automation in the abstract but which function is automated and what human task remains [3]. Call this person the auditor, not the form-filler but the one who asks, every week, whether the evidence still supports the trust being claimed. When they stop asking, you get accuracy theatre: the dashboard is green because the easy metric is green, and the escaped failure is missing because nobody built the path that would measure it.

Rung discipline is the third. Every ambitious agent has two rungs on the autonomy ladder: the rung it was approved at, and the rung it operates at after people get used to it. Someone must own the gap between them. If a support agent starts as a drafting tool and later receives live tool credentials, that is not a minor implementation detail. If a per-action approval queue becomes bulk approval, that is not the same control wearing the same name. If a bounded tool agent starts composing its own tool chains inside a broader goal, the system has crossed from an enumerated action space into a runtime-composed one. The owner here is the governor, with enough authority to say, out loud: this kind of action may climb, this one must demote, this one stays in preview mode, and this one needs a new owner before it acts again. Without that authority you get silent migration; nobody decides to move up a rung, the work volume decides for them, and no meeting is ever held.

Runtime evidence is the fourth. Compliance is not a PDF and governance is not a meeting. In an agentic system the evidence either appears while the system runs, or it arrives too late to govern the system that produced it. Runtime evidence means that traces, approvals, action

IDs, tool calls, memory writes, policy checks, exception queues, rollback receipts and owner names are all captured as part of the work itself, not because regulators love logs, but because a system that cannot reconstruct its own consequential actions cannot learn from them, defend them, or stop repeating them. The person who holds this discipline is the operator, and the operator is not just someone watching dashboards. The operator is the person who can turn a strange run into an accountable record before the anomaly becomes a story. The failure mode is ceremonial governance: the team has principles, review boards, launch docs and audit binders, and the one run that hurt someone has none of the fields needed to answer what happened.

Human attention closes the set. "Human in the loop" is not a magic phrase. It is a labour design, and a fragile one. If the human sees too much, they stop seeing. If they see too little, they cannot intervene. If they see only polished summaries, they inherit responsibility without context. If they are asked to approve a hundred low-risk actions and one high-risk action in the same queue, the queue has trained them to miss the one that matters. The empirical frame is Lee and See's trust-calibration model: trust mediates reliance, and miscalibrated trust, whether over-trust or under-trust, is the mechanism by which a capable system produces poor joint human-machine performance, so the design target is appropriate reliance, not "human in the loop" as a slogan [11]. This discipline belongs to the manager, who protects attention as a production resource: limiting review volume, separating action classes, rotating reviewers, preserving skill, and asking whether human review is actually catching failures or merely producing a comforting artefact. Consider a reviewer signing off a system that is right, say, ninety-something times in a hundred (a made-up number, used only to show the trap). The system is right so often that the reviewer slowly stops reading and becomes a signature rather than a safeguard. Only a manager who is watching the reviewer, not just the model, will catch that drift before the signature is all that is left.

The Tuesday morning dashboard

Five disciplines are five things to remember, and five things to remember on a busy morning is four too many. So they have to live on one screen. Return to that Tuesday from the opening: agents in production, the dashboard green, the first real meeting already moving past it. The point of the dashboard is not to display sophistication. It is to ask five questions before the day runs away with you, and to put a name beside each answer.

The five questions are plain, and they are the whole of the morning. Which component changed? Which error escaped? Which action class changed rung? Which run cannot be reconstructed? Which human review process has stopped giving signal? Each question is one discipline turned into a start-of-day signal. Each carries two names, because responsibility splits cleanly into two roles that organisations chronically merge and should not: the person who does the work, and the person who answers for the outcome.

Each question maps to one discipline and splits its ownership in two. The platform architect is responsible for *which component changed?*, with the CTO or technical owner accountable. An

evaluation lead is responsible for *which error escaped?*, with the head of reliability accountable. A product governor is responsible for *which action class changed rung?*, with the business owner accountable. An observability engineer is responsible for *which run cannot be reconstructed?*, with the compliance lead accountable. The operations manager is responsible for *which human review process has stopped giving signal?*, with the functional leader accountable.

Written as a worksheet, it is one row per discipline and five columns: the discipline, the responsible person, the accountable person, the evidence that turns the row green or red, and the red-row action, the specific move triggered when the answer comes back wrong.

DISCIPLINE	RESPONSIBLE	ACCOUNTABLE	EVIDENCE	RED-ROW ACTION
Layer accounting	Platform architect	CTO / technical owner	New or changed components since last review	Freeze promotion until the change is named and tested
Error accounting	Evaluation lead	Head of reliability	Escaped-failure rate, repeat-run instability, severity-weighted incidents	Drop the action class a rung or narrow its envelope
Rung discipline	Product governor	Business owner	Current rung by action class, any unapproved rung change	Demote or split the action class; update contract, docs, dashboard
Runtime evidence	Observability engineer	Compliance lead	Traces missing fields, actions without trace IDs, open rollback receipts	Stop autonomous execution for the affected classes until evidence is complete
Human attention	Operations manager	Functional leader	Reviewer load, missed exceptions, queue mix, escalation latency	Reduce volume, separate queues, add sampling, or lower the rung

That is the whole instrument, and it fits one screen. That assignment is deliberately plain, and it does not require a large company. In a small team one person may hold several of the responsible roles, and that is fine. Combine the people if you must; do not combine away the responsibilities. The dashboard exists to prevent the single most common sentence in an agentic-system postmortem: everyone thought someone else owned it.

A reader new to operating-model language may want one gloss before the columns make full sense. The dashboard uses "responsible" and "accountable" as separate columns on purpose. Responsible is the person who does the work; accountable is the single person who answers for the outcome and signs it off, even if they did not do the work themselves. The dashboard names both on every row because conflating them is the single most common failure of agentic operating models: an organisation that lists only one name for every discipline either has the same person doing and judging the work, or has no one judging it at all.

One disambiguation rule keeps the worksheet from becoming a labelling dispute. When two readers would name different people for a row, and they will, since in a real org the platform architect and observability engineer may be one person, or the head of reliability and the compliance lead may both claim error accounting, the tie-breaker is the red-row action. Whoever can actually execute that specific move (freeze the promotion, drop the rung, stop execution) is responsible; whoever signs that it was right to is accountable. If no one can execute the red-row action, that is the finding, not a naming quarrel: the row is unowned, and you have learned the most important thing the worksheet can tell you.

If all five rows are green, ship with care. If one is red, the answer is not, by reflex, "try a better model." The answer is to find the owner and change the system. A red row is not paperwork. It means a person may be relying on an answer no one checked, a transaction no one can explain, a memory no one can erase, or a conversation no one is watching with enough care.

The five disciplines are not a competitor to NIST's AI Risk Management Framework, the standard risk framework most teams already know. They are that framework, named in operator vocabulary and bound to the artefacts a dashboard can actually display, so a reader who has built a risk profile already has most of this scaffolding under a different name. The framework names four functions, *Govern*, *Map*, *Measure* and *Manage* [15], and most of the crosswalk is the obvious one-to-one you would guess: layer accounting is *Map*, rung discipline is *Govern*, the four chained reliability numbers are the metrics *Measure* asks an organisation to track, and the trace store is the evidence *Manage* runs on. The one mapping that is not obvious is human attention, because it refuses to sit in a single function: queue design is *Govern* policy, while reviewer load is *Manage* response capacity, so the discipline that is easiest to dismiss as a slogan is the one that touches both halves of the framework at once. That is the tell that this is the same object, not a rival to it.

The outside world has stopped accepting "the model answered"

The reason to do all of this is not only operational neatness. It is that public institutions have stopped accepting "the model answered" as the end of the question, and so should you.

Several of the matters here are live, and they should be handled with restraint, as procedural records and allegations rather than findings of liability. In September 2025 the United States Federal Trade Commission opened an inquiry into chatbots that act as companions, formally ordering several companies to answer questions about how their products are tested,

monitored and age-restricted [6][7]. An inquiry is a regulator asking questions, not a regulator announcing a verdict. Alongside it, companion-chatbot litigation is moving. In one Florida case a federal court issued a 2025 order that allowed parts of a claim to proceed [4]; the docket has since moved on, so that order is a snapshot of one procedural stage and not a settled outcome [5]. In a separate case the defendant company has filed its answer contesting the allegations [8]. And a plaintiffs' release from two advocacy law groups sets out a further set of allegations as one side's claims [9]. Those threads are different kinds of evidence (a court order, an information order, a defendant's answer, a press release) and I am not stacking them. But they point one way. Courts and regulators are now asking product-level questions about safeguards, testing, monitoring, age controls and design choices. "The AI did it" is no longer an accountability posture an institution will accept, and stewardship is the discipline of making sure your own organisation does not accept it either.

The same pressure lands even where there is no AI statute behind it. In Canada the previous federal AI bill died in January 2025 and has not been reintroduced; in its place, federal governance of generative AI now rests on a voluntary code that commits more than forty signatories, including Cohere, IBM, Mila, the Vector Institute and CIBC as the first major Canadian bank to sign, to six outcomes: accountability, safety, fairness, transparency, human oversight, and validity and robustness [12]. A voluntary code is not a statute, and the absence of binding federal law is itself part of the operating picture. But the six outcomes map onto the five disciplines almost without translation, and any deployer who signs the code has already conceded that "the AI did it" is not the end of the conversation.

What is striking is that three governments reach for three different instruments and all three converge on the same duty: a statute, a voluntary code and deployer-side guidance, each placing the answer on whoever fielded the system rather than whoever built the model. Where Canada reaches for a voluntary code, the EU reaches for a statute and the UK for deployer-side guidance, and they arrive at the same destination from three directions. In May 2026 the European Commission's AI Office opened consultation on draft guidance for the transparency rules in the EU AI Act, the rules on disclosing when a person is dealing with AI or with AI-generated content, with the obligations themselves applying from August 2026 [13]. That guidance is still draft, so the operative regime is the statute and the consultation reading; implementation specifics may shift before the August date and again as enforcement begins. In the UK, the data-protection regulator has published guidance on foundation-model deployment that frames transparency, contestability and meaningful human review as duties of the deployer rather than the vendor [14]. The UK has deliberately not legislated an AI Act of its own; instead, its AI Security Institute runs pre-deployment evaluation of frontier models and publishes its findings as a non-statutory deployer-side signal that sits beside the regulator's guidance [16]. Neither instrument names the five disciplines, but each reads through to them: transparency presumes layer accounting, meaningful human review presumes human-attention design, and a deployer-side duty presumes a named owner who can answer for the system.

Vendors are moving in a related direction, bringing governance language closer to the model itself, and this is the part of the picture worth dwelling on, because it is new. In January 2026 Anthropic published a constitution for Claude: a written document stating, in priority order, the values the model is meant to weigh when those values pull against each other. The stated hierarchy puts being broadly safe first, being broadly ethical next, then compliance with Anthropic's own guidelines, and then being helpful, with the lower priorities yielding to the higher ones when a real situation forces a choice [10]. What is interesting is not that a vendor has values, but that the vendor has made the *ranking* explicit and public, so that a buyer can read how the model is intended to resolve a conflict rather than guessing. That is a real advance in legibility. But notice exactly what it does and does not reach. The hierarchy lives inside the model. It does not decide your refund authority, your deletion permissions, your escalation staffing, your trace schema, your audit schedule or your reviewer load. It does not know the action classes you have wired into production, and it cannot know which manager is accountable when an approval process quietly stops working. A vendor advocate will push back: a capable enough model could be told your action classes at runtime and bind itself. But even a model handed those action classes cannot hold the bound that matters, because that bound belongs to someone the model cannot be, a person with the standing to revoke its credentials when the model's own judgement is the thing in question. The constitution governs what the model intends; it cannot govern what the model is permitted, because permission is a property of the system, not of the reasoner. The agent does the work. The system does the trustworthiness. The human does the stewardship.

The objection that lands hardest here is one of scale. A five-discipline operating model can sound like enterprise furniture, and a fair reader will ask what it has to say to a three-person startup or a product team trying to learn something before the market window closes. The answer is not to make small teams pretend they are banks. It is to scale the ceremony down and keep the responsibilities intact. In a small team the architect and the operator may be the same person; the auditor may be a weekly review hour; the governor may be the product lead with one written rule, that no action class climbs a rung without a recovery path; the manager may be whoever controls the review queue. That can be enough early on. But if nobody owns a discipline, the discipline is simply absent. If no one has authority to demote an action class, the autonomy grant is unenforced. Small teams do not get a waiver from physics. They get a smaller checklist.

But the harder objection is not about size at all. It is that an operating model can itself become theatre. A team can name an architect, an auditor, a governor, an operator and a manager, hold the weekly review, keep the dashboard, and still ship harm, because naming an owner is not the same as giving that owner the standing, the time and the information to act. A governor with no authority over the launch date cannot demote anything. An auditor with no access to escaped-failure data audits a fiction. This is the failure the five disciplines are most likely to slide into, and the only honest defence against it is a test applied to each named owner.

> **The Standing Test.** For every named row on the dashboard, ask two questions of the named owner:
>
> 1. **Can this person actually stop the system this week?**
> 2. **Have they been given what they would need to know that they should?**
>
> If the answer to either question is no, the row is not owned. It is decorated. The framework does not protect you from that. It only tells you where to look for it.

When the answer is no, the remedy is not to fix the named owner but to raise the row one level. An unenforceable governor is not a governor problem; it is an accountability gap that belongs to whoever can move the launch date. Naming that person is the one move the test forces upward, instead of leaving the gap stranded in a row no one in the room has the standing to close. That is also the answer to the hardest hostile read of the test, the read that says a CTO who already lacks the standing to halt a launch will equally lack the standing to enforce the test. They might. But the test does not ask them to enforce it alone; it asks them to name, in writing, the person who can, and a named gap at the top of an organisation is a different object from an unnamed one.

A fair reader will want to know what all this evidence does and does not prove. The broad human-factors claim is old and durable: automation changes the human role rather than erasing it, and trust calibration, not headcount, is the design target [2][3][11]. The public documentation around deployed agents is thin and uneven, at least in the current index snapshot, and the exact counts need refreshing because the index is a live project [1]. Regulators and courts are asking product-level questions about companion systems, safety testing, monitoring and design, and the court order, the FTC inquiry and its information order support that narrow claim [4][6][7]. At least one major vendor has made a model-governance hierarchy public, which tells you governance is becoming explicit; it does not tell you any vendor has solved the problem outside the model, and nobody has [10]. What it does not tell me is also worth saying plainly: how the live cases will resolve, the private reliability numbers for most deployed agents, and whether the five-discipline model is enough for the highest-stakes companion deployments without additional domain-specific controls. Those gaps are real. They are not reasons to wait. They are reasons to build the evidence while the system runs.

If the dashboard is the artefact, here is the path to standing it up, and it is four weeks, not a program. In week one, take inventory: list every agent in production, every tool each one can reach, and the named owner of each. Most teams discover at this step that the list is longer and the owners vaguer than anyone expected. In week two, run the reliability equation on a single action class, how often the system errs, how often that is missed, how often a missed error escapes, how severe the result, and resist the urge to do all of them at once; one honest number beats five guessed ones. You cannot compute an escaped-failure rate without a labelled sample, so week two's real task is to pull thirty completed runs of that one action class and have a human adjudicate each as correct, caught-error or escaped-error. That adjudicated

thirty is the honest number, and it doubles as the audit trail proving you measured rather than guessed. In week three, request three traces for that action class: one successful run, one failed run and one near-miss. If any of the three cannot be produced in full, you have found a runtime-evidence gap before it found you. In week four, build the Tuesday dashboard from the worksheet above, assign the ten names, and demote one action class that is currently running unmanaged. Demoting something real on the first pass matters: it proves the dashboard has teeth, and a dashboard without teeth is the theatre this essay warns against.

This capstone is also where the series instruments come together. The framework crosswalk figure that accompanies it maps each problem a reader faces (what did we deploy, where did it break, how bad is the risk, who acted, may it act, who owns it) to the one instrument that answers it, so the stack, the nine layers, the reliability equation, the autonomy ladder and the dashboard read as views of one system rather than a stack of unrelated slides.

— FIGURE S.2

One system, many instruments

The series is not fourteen competing frameworks. It is one system, and each question has the instrument that answers it.

	THE INSTRUMENT THAT ANSWERS IT	ESSAY
What did we actually deploy?	The stack	E01
Where did it break?	The nine-layer map	E04
How bad is the risk?	The reliability equation	E05
What can it touch?	The permissions triangle	E09
What text entered the system?	The 4P Gate	E10
What can steer it at runtime?	The four-corner defense	E11
What state persists?	The memory lifecycle	E13
Why did it stop?	The stopping rule	E14
Can we inspect a run?	The trace tree	E15
Did the meaning hold?	The semantic grid	E16
How do we weigh the evidence?	The three witnesses	E17
Who acted?	The identity chain	E18
Can we prove compliance?	The runtime primitives	E19
May it act on its own?	The autonomy ladder	E21
Who owns it on a Tuesday?	The Tuesday dashboard	E22

• CONCEPTUAL MODEL ARCHITECTING THE AI COWORKER

One system, many instruments. Every question the series teaches you to ask, and the instrument that answers it.

Enough about the program. Here is one morning. Back in the room the series opened with: a support lead, Tuesday, one strange overnight run, the refund-drafting agent has issued a store credit to a customer flagged as a chargeback risk. Name the stack: the support-refund agent

at the version that shipped last week, with the policy-retrieval tool, the refund-quote tool and the customer-record reader. Bound the authority: rung 3 on store credit under the local low-value cap in enumerated reason codes, with chargeback flag set to refuse. Preserve the evidence: the trace ID resolves to the policy version read, the chargeback flag recorded as present on the customer record, and the absence of any corresponding flag-check event in the action log (the check did not merely fail, it was never invoked) alongside the approval-screen event that the rung-3 policy required and that no log entry shows. The trace proves the non-event by what is missing where the schema demands an entry. Name the owner: the Head of Support Ops, who has the authority to demote the class to rung 2 by lunchtime and the trace evidence to defend the demotion in writing. Four moves, one run. They are not a slogan; they are what the meeting needs by the time the second coffee is cold.

There is a Tuesday morning in the future when an architecture you helped design will be running, quietly, in the background of someone else's work. They will not see the components. They will not see the rungs, or the traces, or the names beside each discipline. They will see a coworker, and they will trust it the way people trust colleagues. The series began by taking apart that exact trust, the illusion of the coworker who is a stack, and everything between then and now, the expected-harm equation, the autonomy ladder, the runtime evidence, the five disciplines, was the work of earning it back honestly. The series opened in a room saying "the AI did it": four words, a small shrug, an investigation closed inside its smallest pronoun. The whole twenty-one essays between that room and this one have been the work of replacing the pronoun. We close with the same room saying four different things.

> **From "the AI did it" to:**
>
> 1. **Name the stack.** Here is the stack we deployed.
> 2. **Bound the authority.** Here is the authority it acted under.
> 3. **Preserve the evidence.** Here is the evidence of what it did.
> 4. **Name the owner.** Here is the named owner who answers for the outcome.
>
> Four words out. Four sentences in. That is the sentence an architected team can actually say.

The instruction the whole series has been walking towards is a change of verb. Stop delegating to a thing you have decided to call a coworker, because delegation hands work to someone you trust and then stops looking. Start architecting the team that does the work, the components, the owners, the rungs, the evidence, the attention, each with a name beside it and each able to act.

Carry This Forward

One move now, and the move it sets up.

Do this week: Put the five disciplines on one screen, layer accounting, error accounting, rung discipline, runtime evidence and human attention, and beside each write the person with the authority, the time and the information to stop, demote, redesign or repair the system. A blank row is not an admin gap; it is an unmanaged risk. A row with a name but no real authority is worse, because it looks managed. Then run the Standing Test on every name, and demote one action class that is currently running unmanaged, so the dashboard ships with teeth on its first day.

What it sets up: The work ends where production begins. Trustworthy agents are not hired like coworkers; they are architected like teams. Fill the operating model before the next agent earns more authority. Name the stack, bound the authority, preserve the evidence, name the owner. Then ship.

Three regulatory lenses US · EU · UK

Operating questions, not legal advice. The frameworks stay the same; the regulator changes.

US Can your Tuesday review answer the five questions through NIST RMF, FTC, state liability, sector supervision and procurement discipline?

EU Can your Tuesday review answer the five questions through AI Act, GDPR and product / cyber duties?

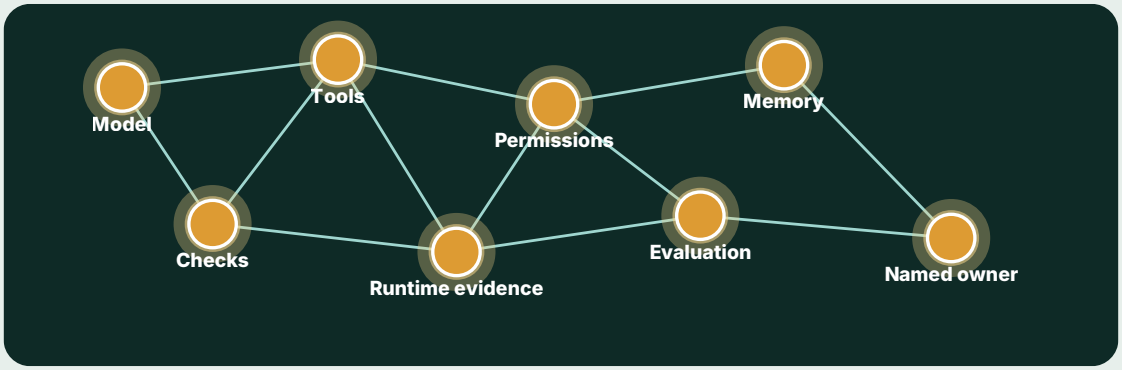
UK Can your Tuesday review answer the five questions through sector regulators, ICO, NCSC and the DSIT AI Cyber Security Code?

THE STACK SO FAR

E22 · Essay 22 of 22 complete · Arc V: Operating model

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	V **Operating model** ESSAY 3 OF 3



stack you have built across the series

the whole stack is in scope

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

Series complete. You can now architect the team. Evaluation, autonomy, ownership, operating discipline.

You have just added.

The Tuesday Dashboard and framework crosswalk

You can now operate the whole AI-coworker team through named disciplines.

This closes the series. The next move is yours.

← PREVIOUS
E21 · The Autonomy Ladder

Essay 22 of 22 complete

NEXT →
This closes the series

References

Reference links for sources cited in this essay.

1. **MIT AI Agent Index**
   MIT authors
   https://arxiv.org/abs/2602.17753

2. **From here to autonomy: Lessons learned from human-automation research**
   Endsley, M. R.
   https://doi.org/10.1177/0018720816681350

3. **A model for types and levels of human interaction with automation**
   Parasuraman, Sheridan, Wickens
   https://pubmed.ncbi.nlm.nih.gov/11760769/

4. **Garcia v. Character Technologies order**
   M.D. Fla. / court PDF via Ars Technica CDN
   https://cdn.arstechnica.net/wp-content/uploads/2025/05/Garcia-v-Character-Technologies-Order-on-Motion-to-Dismiss-5-21-25.pdf

5. **Garcia v. Character Technologies docket**
   DocketAlarm / M.D. Fla.
   https://www.docketalarm.com/cases/Florida_Middle_District_Court/6--24-cv-01903/Garcia_v._Character_Technologies_Inc._et_al/

6. **FTC inquiry into AI chatbots acting as companions**
   FTC
   https://www.ftc.gov/news-events/news/press-releases/2025/09/ftc-launches-inquiry-ai-chatbots-acting-companions

7. **FTC AI Companion Chatbot 6(b) Order**
   Federal Trade Commission
   https://www.ftc.gov/system/files/ftc_gov/pdf/AICompanionChatbot6%28b%29Order.pdf

8. **Raine v. OpenAI Answer PDF**
   OpenAI / court filing via Ars Technica CDN
   https://cdn.arstechnica.net/wp-content/uploads/2025/11/Raine-v-OpenAI-Answer-11-25-25.pdf

9. **SMVLC/TJLP lawsuits release**
   Tech Justice Law Project / SMVLC
   https://techjusticelaw.org/press-releases/social-media-victims-law-center-and-tech-justice-law-project-lawsuits-accuse-chatgpt-of-emotional-manipulation-supercharging-ai-delusions-and-acting-as-a-suicide-coach/

10 **Anthropic Constitution**
Anthropic
https://www.anthropic.com/constitution

11 **Trust in automation: Designing for appropriate reliance**
Lee, J. D., & See, K. A.
https://doi.org/10.1518/hfes.46.1.50_30392

12 **Voluntary Code of Conduct on the Responsible Development and Management of Advanced Generative AI Systems**
Innovation, Science and Economic Development Canada (ISED)
https://ised-isde.canada.ca/site/ised/en/voluntary-code-conduct-responsible-development-and-management-advanced-generative-ai-systems

13 **Draft guidelines on Article 50 transparency obligations under the AI Act**
European Commission AI Office
https://digital-strategy.ec.europa.eu/en/consultations/consultation-draft-guidelines-transparency-obligations-under-ai-act

14 **ICO guidance on foundation models and generative AI**
Information Commissioner's Office (UK)
https://ico.org.uk/about-the-ico/what-we-do/our-work-on-artificial-intelligence/

15 **Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1: Govern, Map, Measure, Manage functions**
National Institute of Standards and Technology
https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf

16 **UK AI Security Institute (formerly AI Safety Institute), Department for Science, Innovation and Technology, AISI work program and frontier-model evaluation reports**
UK Department for Science, Innovation and Technology
https://www.aisi.gov.uk/

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