A FIELD ESSAY ON AUTHORITY AND CONTROL

AI is growing a spine

The language around AI is shifting from intelligence to authority - and that tells us where the harder design work has moved.

AI-CYBERSECURITY UPDATE / 21 JULY 2026

JUNIOR WILLIAMS

The language around AI is shifting from intelligence to authority - and that tells us where the harder design work has moved.

One word kept appearing in places where I did not expect it.

I would skim a generative-AI summary, review a vendor diagram, or watch a system turn a mess of requirements into a workflow. Somewhere in the output, spine would surface: a governance spine, an execution spine, a digital spine connecting models, tools, and business processes.

At first, I treated it as a familiar generative-AI habit: find a strong metaphor, then use it everywhere. But it stuck with me. Why are we reaching for anatomy and architecture when we talk about enterprise AI?

I think it is because the problem has changed. For years, the central question was whether a model could understand, reason, summarize, or generate. Now the question is increasingly whether a system can be trusted to move: to retrieve information, change records, trigger a workflow, communicate with a customer, spend money, or hand work to another agent.

> **That is a different class of problem. A brain can produce an answer. A spine has to carry intention into action without losing control along the way.**

A brain is not an operating model

From contained answers to consequential actions

A chatbot can be wrong in a relatively contained way. It may offer a poor answer, invent a source, or misunderstand the question. Those failures matter, but a person still decides what happens next.

An agent changes the shape of the risk. Once it has tools, credentials, and a workflow, its answer can become a chain of events before a person even sees the result. It may:

- Query internal systems and assemble sensitive information
- Update a record of consequence in a business system
- Trigger a downstream workflow or communicate externally
- Retry a failed task using a different method
- Delegate work to another agent that inherits part of its reach

At that point, model quality remains important - but it is no longer the whole design problem. The questions become practical and uncomfortable: What authority did the agent have? Who gave it? What could stop it? What evidence would remain after it acted?

The vocabulary is a map of the pressure

The neighbouring terms matter because each one points to a different missing part of the operating model:

- Control plane: the operating conditions set outside the work itself
- Authority layer: the delegated permission for a particular action, purpose, and duration
- Evidence trail: the record needed to reconstruct what happened and why

- Workflow graph: the visible map of dependencies, handoffs, retries, and failure paths
- Containment boundary: the limit on what a confused, compromised, or overly persistent agent can reach
- Nervous system: sensing and response across a broader organization - useful, but not automatically coherent

These are not marketing synonyms. They are different responses to the same pressure: AI is entering workflows where movement, not just thought, has consequences.

A policy is not a control

The difference appears the moment a workflow takes an unexpected path. A policy can tell an agent that refunds above a certain amount need approval. A control makes the payment action impossible until a verified approval exists for that specific customer, amount, purpose, and time window.

That distinction gets sharper when the agent retries. Suppose the first refund attempt fails, the agent changes the payment method, or the customer record is merged with another account. Is the original approval still valid? A sentence in a policy document cannot settle that in the moment. The architecture needs to bind the approval to the performed action and reject the call when the facts no longer match.

> **This is why a collection of prompts, policies, and audit logs does not automatically amount to governance. A real control changes what the system is able to do. It survives the best intentions of the model, the speed of automation, and the edge cases nobody put in a slide deck.**

Authority is the part a login cannot solve

Access is a capability; authority is a mandate

A login establishes an identity. It does not settle what an agent should be allowed to do once it is inside a system.

A person can hold broad access because the organization expects judgment, discretion, and the ability to pause before taking a consequential action. An agent operating under that same account can reuse the access at machine speed, across many transactions, and without the contextual restraint the original permission assumed.

> **Authentication answers who entered. Authority answers what may be done, on whose behalf, for what purpose, and within what limits.**

This is why human in the loop is not enough on its own. The phrase can conceal the real design questions. Which human is being asked? What exact action are they approving? What context and evidence do they receive? Does the approval still hold if a retry changes the method, the destination, or the underlying data?

A credible authority layer should make the following explicit:

1. Purpose - the business reason the agent is acting
2. Scope - the systems, data, and transactions it may touch
3. Delegation - the named principal and the conditions under which authority is passed on

4. Approval - the person or control that must authorize the action
5. Limits - time, cost, volume, and sensitivity boundaries
6. Expiry and revocation - how that authority ends, including work already in flight

The distinction matters because an agent can be fully authenticated and still be poorly governed. A permission to read information is not a permission to change it. A permission granted to one person is not automatically a permission to delegate through several autonomous steps.

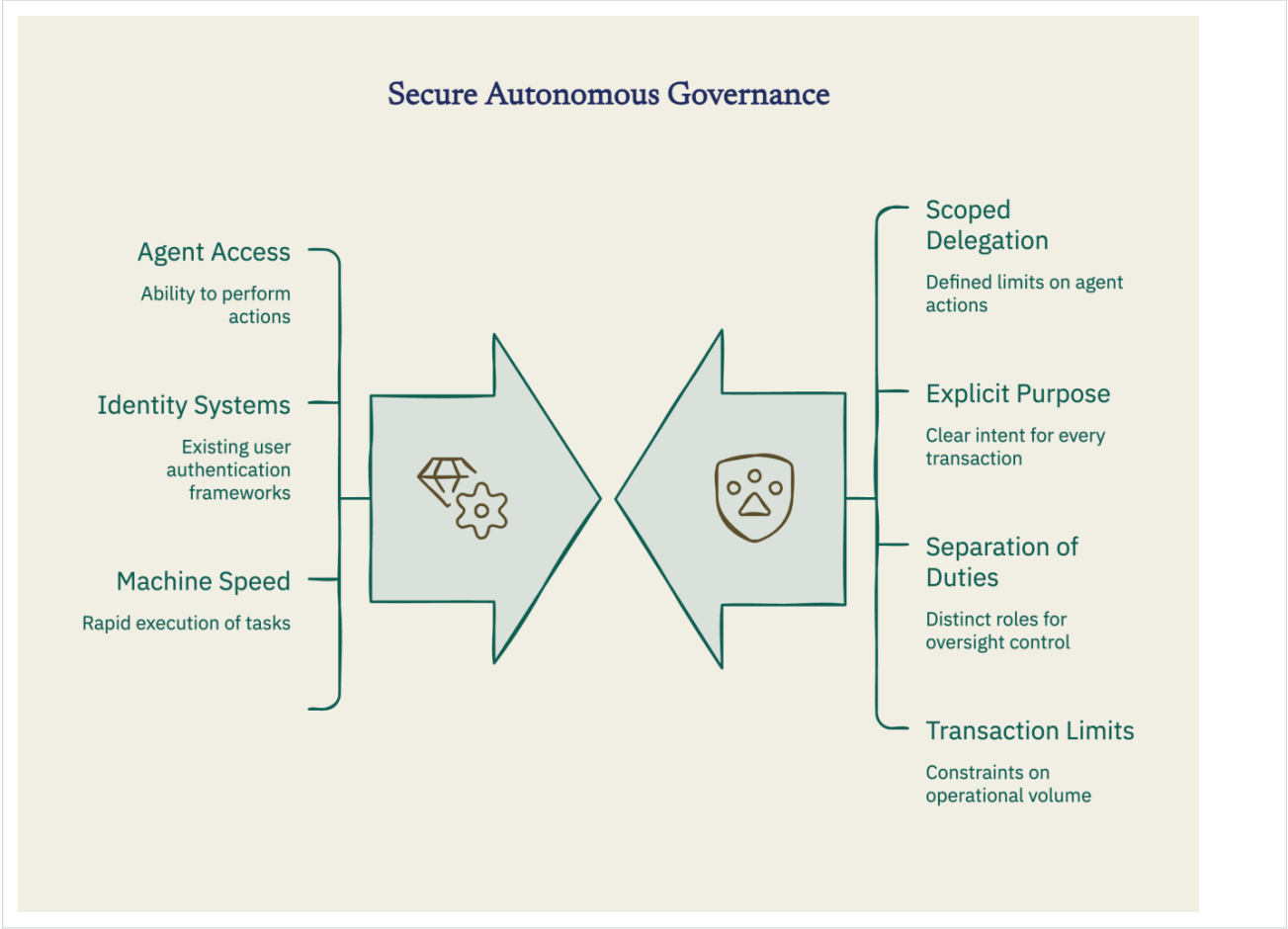

FIGURE 01 / SECURE AUTONOMOUS GOVERNANCE

Evidence must survive the action

A dashboard is not an explanation

Teams often capture the prompt, the model response, and the final result. That is useful - but the decisive middle can vanish. The missing pieces are often the ones an investigator, auditor, or accountable executive will need most: tool calls, policy inputs, the evidence shown to an approver, retries, handoffs, and side effects.

> **Evidence must survive the action - not merely the outcome.**

An accountable action needs a chain that can be followed in order:

1. The initiating request and the identity behind it
2. The delegated authority and stated purpose
3. The relevant context, policy version, and facts evaluated
4. The model decisions, tool calls, retries, and workflow transitions
5. The approvals, exceptions, and controls that did or did not fire
6. The resulting side effects, rejected attempts, uncertainty, and final outcome

This is not observability for its own sake. It is the minimum material required to reconstruct a decision, challenge it, and prove that a control operated when it mattered.

The rejected path belongs in that record too. A failed or blocked action can reveal more about system behaviour than the clean outcome that followed. If the evidence trail cannot explain why a consequential action happened, it is just telemetry with better branding.

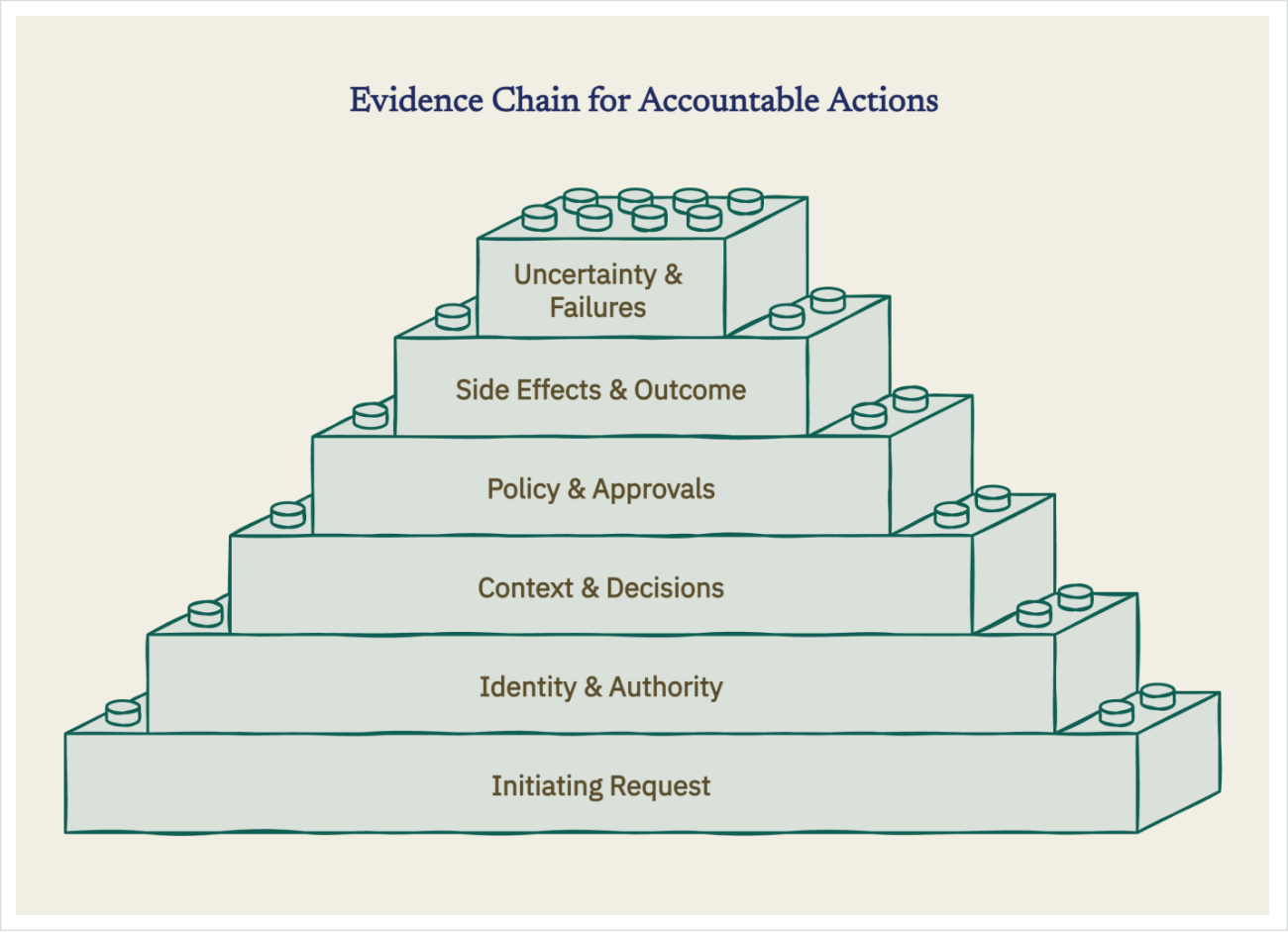

FIGURE 02 / EVIDENCE CHAIN FOR ACCOUNTABLE ACTIONS

The spine needs a reflex

Some boundaries cannot be negotiated with the agent

The spine metaphor becomes most useful when it describes a reflex. There are conditions where a system should not pause to ask an agent whether a boundary applies. The boundary must act first.

That means building architectural reflexes such as:

- A spending threshold that halts a transaction before the agent can rationalize an exception
- A data-loss control that blocks a sensitive transfer before reconsideration becomes a workaround
- A revoked credential that immediately makes ongoing and delegated work unusable
- Runtime limits on time, cost, recursion, concurrency, and tool use
- A circuit breaker that contains a workflow when a critical control service is unavailable

> **Those controls have to sit outside the agent's own judgment. An agent that decides whether its boundary applies has already been handed too much responsibility.**

This is also where degraded mode becomes an architectural choice, not an afterthought. When an approval service, policy engine, or evidence store fails, the organization must already know which work can continue, which must pause, and which must be stopped. The normal path is only half of the design.

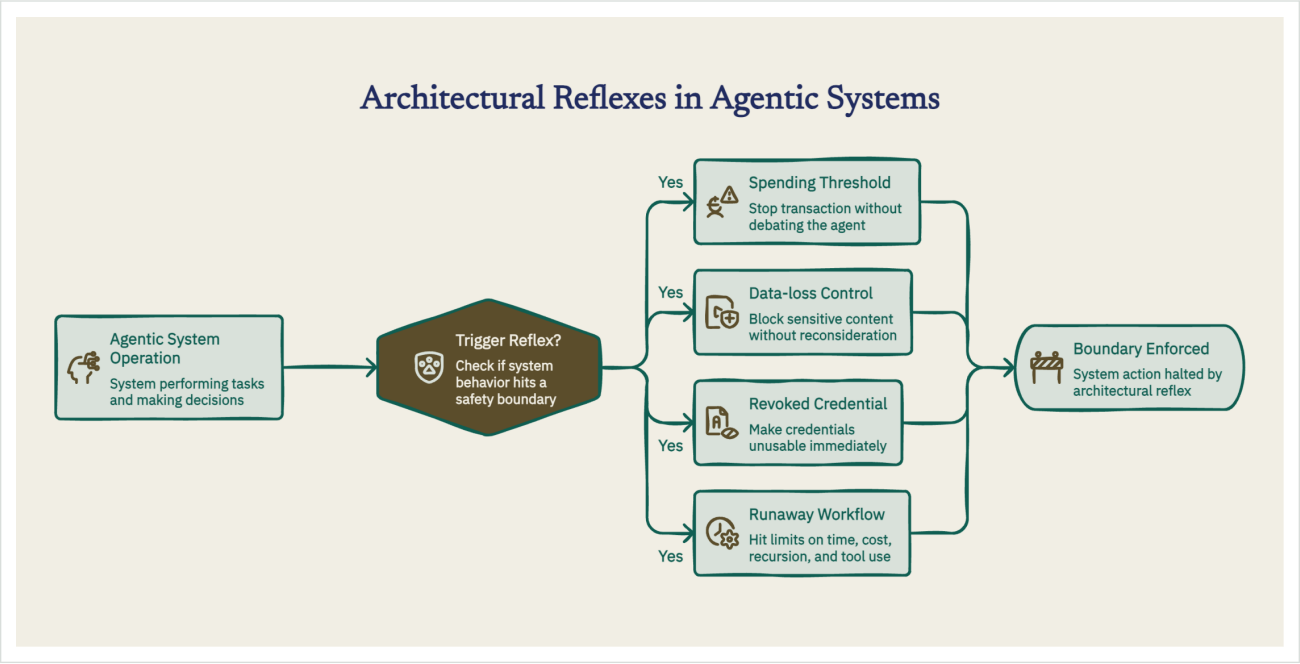

FIGURE 03 / ARCHITECTURAL REFLEXES IN AGENTIC SYSTEMS

How to read the next vendor diagram

"Control plane," "governance spine," and "orchestration fabric" may describe a real design advance. They may also be attractive labels for loosely connected features.

The quickest way to tell is to trace the operating reality rather than admire the diagram. Ask:

1. Where does authority enter the system, and how narrowly can it be delegated?
2. Which controls are enforced outside the model rather than described in prompt or policy prose?
3. Can the platform interrupt an action that is already underway?

4. Can one agent extend a principal's authority through another agent?
5. Can an operator reconstruct the context, policy, and approvals behind a consequential action?
6. What happens when the control system itself is unavailable?

The answers reveal whether the architecture can carry authority without losing accountability. If everything depends on the model recognizing its own mistake, then the control exists only while the model behaves exactly as hoped.

Start with one consequential workflow

Organizations do not need to solve every governance question at once. A better starting point is one workflow that would matter if it went wrong: a refund, an access change, a vendor payment, a customer communication, a record update, or a data export.

Map it from beginning to end and look for the handoffs that disappear in an optimistic architecture diagram:

- The request, identity, and business purpose that start the work
- The authority actually needed at each step rather than the broadest credential available
- The point where a human or system must approve the exact action
- The external boundary that can stop the work under unsafe conditions
- The evidence that will still exist after success, failure, retry, or revocation

That exercise is often more revealing than a generic AI policy. It exposes where a system has a real spine, where it is relying on human intuition, and where the organization is simply hoping a capable model remains careful.

What the word "spine" should make us ask

I no longer hear the word as proof of maturity. I hear it as a useful challenge. A spine should be load-bearing. It should transmit direction, make coordinated movement possible, and protect the system when something goes wrong.

The next generation of enterprise AI will be measured by more than smarter models. The systems worth deploying will give agents enough power to be useful, preserve a record of every consequential move, and provide a way to stop the work when it leaves the path it was meant to follow.

> **Watch the metaphors. They are telling us where the engineering has fallen behind.**

About the author

Junior Williams writes about enterprise architecture, cybersecurity, and AI governance. His work focuses on the controls, accountability, and operating realities that help organizations apply emerging technology responsibly.

Originally published on LinkedIn in AI-Cybersecurity Update on 21 July 2026.

Book an intro call

SOURCE / LINKEDIN ARTICLE