MERIT

Multi-Agent Orchestration and the Coordination Tax: Why Every Agent You Add Makes the Next One More Expensive to Run

The next agent you deploy will not cost what the last one cost. Enterprises have quietly moved from single agents to chained, multi-agent systems, and the risk in those systems does not live inside any one agent. It lives in the handoffs between them, and almost nobody is watching those handoffs.

Authored by Tharun Mathew
Head of Data & AI Solutions, Merit Data and Technology

CONTENTS

Table of Contents

EXECUTIVE SUMMARY

SITUATION

Enterprise AI has moved past the single-agent era.

A pricing agent that used to work alone now calls a market-data agent, which calls a competitor-intelligence agent, which hands its output to a drafting agent, which hands the draft to an approval agent before it ever reaches a customer.

Multi-agent orchestration frameworks have made this trivial to build. By 2026, enterprises running agentic AI in production are chaining an average of 3.4 agents into a single workflow, and that number is rising every quarter.[1]

None of this was planned as a system. It accumulated, workflow by workflow, as teams chained together agents that each worked well individually.

THE AGENT ESTATE IS GROWING

12 AI Agents
Average number run by a company today

20 AI Agents
Expected average by 2027

50%
Operate completely autonomously

Source: Salesforce 2026 Connectivity Benchmark Report

COMPLICATION

The instinct is to treat this as more of the same problem the organisation already has a plan for: govern each agent, log each agent, monitor each agent.

That instinct is where multi-agent programmes go wrong.
A chain of five well-governed agents is not five times the risk of one agent. The number of possible handoff relationships grows roughly with the square of the number of agents, not in a straight line, and only 17% of organisations continuously monitor agent-to-agent interactions at all.[2]

Each handoff is a point where context can be lost, permissions can be inherited incorrectly, and an upstream hallucination can be treated as fact by a downstream agent, with nobody owning the failure because no single agent did anything wrong on its own.
Gartner already attributes a growing share of its forecast that over 40% of agentic AI projects will be cancelled by the end of 2027 to exactly this kind of orchestration failure.[3]

THE AGENT ESTATE IS GROWING

17%
of organisations continuously monitor agent-to-agent interactions.

38%
monitor AI traffic end-to-end across prompts, tool calls and outputs.

The gap is not simply model performance. It is visibility into the interactions that connect autonomous systems.

Source: EY / AIUC-1 Consortium, 2026 [2]

RESOLUTION

You do not close the coordination tax by writing a policy about how agents should behave with each other. You close it by treating orchestration as a **Zero-Trust Control Plane for autonomous agents:** an architectural layer that governs how agents discover, authenticate, delegate to, exchange information with, and hand work to one another.

Orchestration is therefore not simply workflow routing. It is the control plane that determines **which agent is allowed to act, on whose authority, with what context, under what constraints, and with what evidence of execution** at every hop in the chain.
This distinction matters because the coordination layer remains one of the least monitored parts of the agentic stack.

Only **17% of organisations continuously monitor agent-to-agent interactions,** despite the growing adoption of chained autonomous systems.[2] Individual agents may have their own identities, permissions, logs, and behavioural controls, but those controls do not automatically establish trust across the chain.

A downstream agent still needs to know who authorised the request, whether the instruction is within scope, whether the context it received can be trusted, and whether the action it is about to take is consistent with the original task.

Merit approaches this as an orchestration control-plane problem. **Our coordination-ready architecture provides the control layer between autonomous agents,** combining chain-level discovery, delegated identity, scoped permissions, structured handoff contracts, context preservation, and orchestration-level observability.

Rather than attempting to infer governance from individual agent logs after an incident, the control plane establishes the rules, evidence, and constraints at the point where agents interact.

The objective is not to eliminate autonomy. It is to make autonomy **bounded, attributable, observable, and explainable** as it moves across an agentic system.

That is what allows enterprises to scale from individual agents to multi-agent workflows without allowing trust, permissions, or accountability to expand unchecked with every additional agent.

KEY FINDINGS

1. Multi-agent systems fail at the seams, not the agents. Individual agents can pass every quality and safety check an organisation runs, and the system built from them can still fail, because the failure mode lives in the handoff between agents rather than inside any single one.

2. The coordination tax grows combinatorially, not linearly. Adding a fourth agent to a three-agent chain does not add one new relationship to monitor. It adds every new path a task could take between four agents rather than three, and most enterprises are budgeting for linear growth in a problem that grows combinatorially.

3. Nobody is watching the handoffs. Only 17% of organisations continuously monitor agent-to-agent interactions, against 38% who monitor AI traffic end-to-end at all.[2] The coordination layer is the least observed part of the agentic stack and the part carrying the most systemic risk.

4. Failures cascade silently before they surface loudly. A hallucination, a stale data pull, or a permission overreach introduced early in a chain does not announce itself. It propagates as accepted fact through every downstream agent until it produces a customer-facing action somebody has to explain.

5. The fix is a coordination layer, not a bigger model or a longer policy. Closing the gap requires chain-level discovery, delegated identity across hops, structured handoff contracts, and orchestration-level observability. Almost none of this exists in production today.

KEY STATISTICS

12

AI agents is the average number a company runs today, expected to rise to 20 by 2027, yet 50% of these agents operate completely autonomously. (Salesforce 2026 Connectivity Benchmark Report) [1]

17%

of organisations continuously monitor agent-to-agent interactions (EY / AIUC Consortium, 2026)[2]

40%+

of agentic AI projects will be cancelled by end of 2027, with orchestration failures a growing share of the cited causes (Gartner, 2025)[3]

Section 1

Why Multi-Agent Systems Are a Different Problem to Single Agents

The pattern is consistent across the multi-agent deployments reviewed for this whitepaper. A team builds a single agent to do one job well: summarise support tickets, draft an email, pull a pricing figure. It works, so the team adds a second agent to act on the first agent's output. Then a third, to check the second agent's work. Then a fourth, to route the result somewhere.

Nobody sat down and designed a system. The system emerged from a series of individually sensible decisions, and that is exactly why it is dangerous. Each addition passed its own review. The chain as a whole never went through one.

This matters because a chain of agents is not a bigger version of a single agent. It is a different category of system, with a different failure profile. A single agent that gets something wrong produces a wrong output that a human, or a downstream system, can catch and correct. A chain of agents that gets something wrong produces a wrong output that the next agent in the chain treats as ground truth, acts on, and passes further downstream, often with growing confidence rather than growing scepticism, because each agent is reasoning over what looks like clean, structured input from a trusted internal source. The error does not stay where it started. It compounds.

This whitepaper calls this the coordination tax: the accumulated cost, in risk, latency, and governance effort, that a multi-agent system incurs at every point where one agent hands work, data, or a decision to another.

At its technical core, the problem is **Compounding Error Propagation and Hallucination Cascades.** An output that is uncertain when it leaves one agent can become an assumed fact by the time it reaches another.

Consider a simple five-agent chain. Agent A produces an assertion with a 90% confidence score. Agent B receives that assertion as structured input and, unless the handoff explicitly preserves its uncertainty, treats it as an authoritative fact. Agent C reasons over B's output rather than the original evidence.

Agent D incorporates the assertion into a recommendation, and Agent E acts on the recommendation. By the time the information reaches Agent E, the original uncertainty has effectively disappeared.

What began as a **90%-confidence assertion has become an unverified claim masquerading as absolute fact.**

This is how hallucination cascades emerge in multi-agent systems. The problem is not necessarily that every downstream agent hallucinates. Each agent may perform exactly as designed.

The failure occurs because **uncertainty, provenance, caveats, and evidential context degrade as information crosses agent boundaries.** A downstream agent has no reliable way to distinguish a verified fact from an upstream inference unless the orchestration layer explicitly carries that distinction forward.

Modern agentic developer frameworks have made these architectures increasingly straightforward to build. LangGraph, Microsoft AutoGen, and Anthropic's Model Context Protocol (MCP) provide mechanisms for agents, tools, and workflows to discover capabilities, exchange context, and coordinate actions.[11][12]

This represents a major advance in agentic application development, but it also makes the coordination problem more consequential: when the technical barrier to adding another agent is low, the number of opportunities for context loss, permission inheritance, and error propagation increases rapidly.

The coordination tax therefore does not simply represent the cost of adding another agent. It represents the **compounding exposure created by every additional handoff.**

Without structured contracts, preserved confidence and provenance, scoped delegation, and orchestration-level observability, each handoff can convert uncertainty into apparent certainty and an upstream error into a downstream decision.

This is why agent-level performance metrics are insufficient. Measuring whether Agent A or Agent B performs accurately in isolation does not tell an organisation whether the **chain preserves truth, context, authority, and accountability from the first instruction to the final action.** That is the coordination problem the orchestration control plane must solve.

WHERE THE HANDOFF GAP APPEARS

CONTROL-PLANE REQUIREMENT

Preserve provenance + confidence + caveats + authority + execution relationship at every hop.

"Modern agentic architectures don't just run a single autonomous workflow. They chain them. An employee might deploy a research agent that triggers a drafting agent that triggers a send agent. Each step has its own permissions footprint. Each handoff is a potential data exposure." [5]

A sanctioned three-agent pipeline running in production every day carries the same handoff risk as an informal one, except now it is load-bearing for a business process, harder to unwind, and generally assumed by the organisation to be safe because it went through a review. It did.

The review just was not built to catch what happens between the agents rather than inside them.

Section 2

How the Coordination Tax Accumulates in Production

To understand why the coordination tax grows the way it does, it helps to do the arithmetic most enterprises have not done. A single agent has no coordination surface. A two-agent chain has one handoff to worry about.

A three-agent chain has three possible pairwise relationships. A five-agent system, of the kind now common in customer service, sales enablement, and finance operations, has ten possible pairwise paths, and considerably more if agents can call each other conditionally rather than in a fixed sequence.

That mismatch between combinatorial risk and linear budgeting is why programmes that felt manageable at three agents start to feel ungovernable at seven or eight, often within the same financial year.

The agentic thundering herd, previously documented in relation to agent-ready data infrastructure, is where multi-agent orchestration stops being an edge case and becomes the default operating condition.[14]

A customer enquiry that once triggered a single agent can now trigger a triage agent, which spawns research and sentiment agents in parallel, both of which report to a drafting agent, which is reviewed by a compliance agent before a send agent acts.

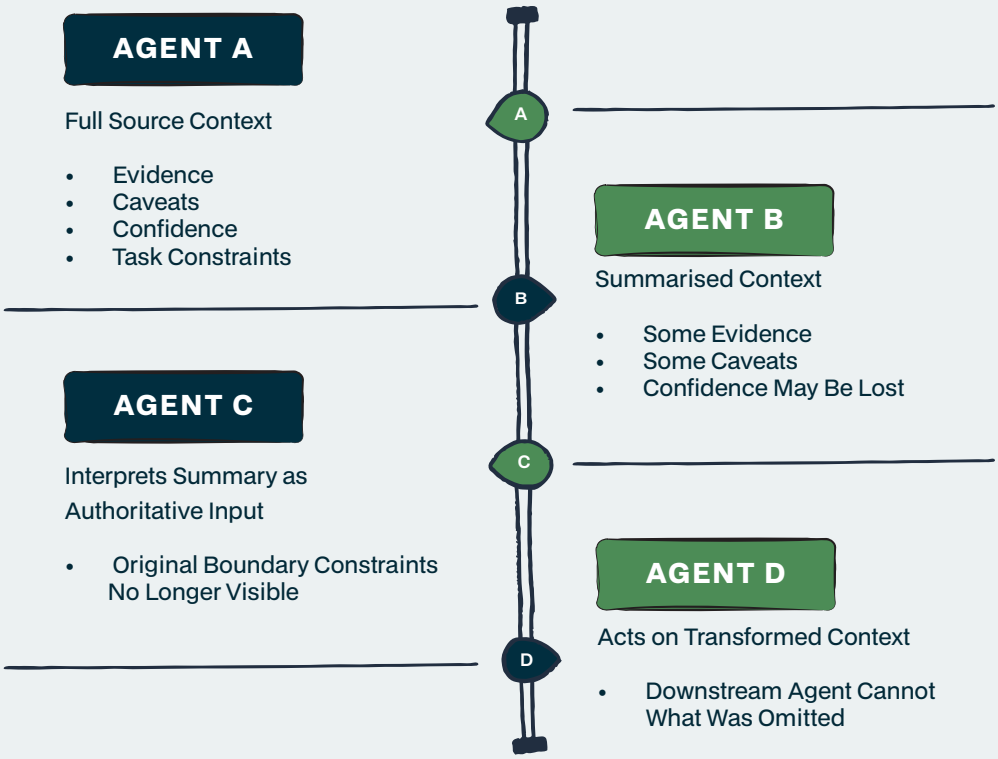
Six agents may be reading from and writing to shared enterprise systems simultaneously, creating both infrastructure pressure and a much larger coordination surface.

But the more dangerous failure modes occur inside the handoffs themselves. One is **Context Window Truncation**. Agents rarely pass their entire working context to the next agent. They typically pass a compressed prompt, summary, or structured subset of the original information.

As chains become longer, this compression can strip out the very details that define the boundaries of an action: exceptions, caveats, confidence levels, approval requirements, or instructions such as do not send externally without human approval.

The downstream agent receives a cleaner-looking input, but a less complete representation of the original task.

CONTEXT DEGRADATION ACROSS A CHAIN



CONTROL-PLANE REQUIREMENT: Schema-enforced handoff + provenance + confidence + constraint preservation.

This creates a particularly dangerous form of **contextual privilege escalation**.

An agent may receive an instruction that is technically valid but missing the constraint that made it safe. Agent B therefore executes what Agent A intended, but without the boundary conditions that Agent A had available.

By Agent D or Agent E, the original constraint may have disappeared entirely while the resulting action still appears legitimate because it originated from another trusted agent.

The identity problem compounds this risk. Multi-agent chains can create **Confused Deputy Attacks and privilege escalation through chain-of-custody** when delegated credentials are passed between agents without being re-scoped at each hop.

For example, Agent A may legitimately hold a broad JWT or OAuth token because its role requires access to customer records, internal pricing data, and workflow APIs. If Agent A invokes Agent B and passes that credential through unchanged, Agent B may inherit permissions that were never intended for its narrower task. Agent B has not compromised the credential or deliberately exceeded its authority.

It has simply become a **confused deputy,** acting with authority inherited from an upstream agent that had a different purpose.

The result is privilege escalation without an explicit privilege-granting event. The fifth agent in a chain can potentially access systems or perform actions because the authority travelled with the request rather than being re-evaluated at each handoff.

In a traditional identity model, the question is what can this agent access? In a multi-agent architecture, the more important question becomes **why does this agent have that access, who delegated it, for what task, and does that authority remain valid at this point in the chain?**

This is why coordination cannot be reduced to routing. Every handoff is simultaneously a . A coordination-ready architecture must preserve the critical constraints that govern the task while issuing narrowly scoped delegation at each hop.

Otherwise, context gets progressively weaker while inherited authority can remain unnecessarily broad, creating a system in which the information required to make a safe decision disappears at exactly the same time that the permissions to act remain in place.

A customer enquiry that used to trigger one agent now triggers a triage agent, which spawns a research agent and a sentiment agent in parallel, both of which report to a drafting agent, which is itself reviewed by a compliance agent before a send agent fires.

Six agents, most of them running concurrently rather than sequentially, all reading from and writing to a data layer that was very possibly designed for one request at a time.

"Agentic systems require infrastructure that supports adaptive, multi-turn interactions in which agents dynamically discover capabilities, share context, and hand off work as tasks evolve."[4]

The identity problem this creates is considerably harder once agents call other agents. An agent acting under a delegated identity, with permissions scoped for its specific task, will often call a second agent using whatever credentials it happens to hold, rather than a freshly scoped, purpose-specific identity for the handoff.

The result is permission creep by inheritance: the fifth agent in a chain can end up operating with access rights that made sense for the first agent's task but bear no relationship to what the fifth agent is actually doing. Nobody granted that access deliberately. It arrived by chain of custody, and in most production environments today, nothing in the identity stack is built to notice, let alone stop it.

Section 3

Why Single-Agent Governance Cannot Absorb a Multi-Agent Estate

Most of the AI governance work enterprises have done over the past two years, including agent discovery, non-human identity, and behavioural monitoring, was built with a single agent in mind. That work is not wasted. It is necessary. It is also, on its own, insufficient for a multi-agent estate, and the reason is structural rather than a matter of degree.

An agent discovery programme, built correctly, will tell an organisation what agents exist, what they are permitted to do, and roughly what data they touch. What it typically will not tell the organisation is which agents call which other agents, in what order, under what conditions, and with what data passed at each hop.

That information, the topology of the agent estate rather than its inventory, lives nowhere in most enterprises today. It exists as tribal knowledge among the engineers who built each individual chain.

Giving each agent its own managed, least-privilege identity solves the problem of distinguishing what a human did from what an agent did. It does not, on its own, solve the problem of distinguishing what Agent A asked Agent B to do from what Agent B decided to do on its own initiative.

A regulator or an internal auditor investigating a wrong customer-facing decision needs to reconstruct the entire delegation chain: who asked whom, with what instruction, on what authority, and whether each agent stayed within the scope of the task it was actually given. Most identity architectures capture the first question well and cannot answer the second at all.

Behavioural monitoring, built to flag when a single agent acts inconsistently with its declared purpose, faces a harder problem applied to a chain: each agent can behave entirely consistently with its own declared purpose while the chain as a whole produces an outcome nobody intended.

A drafting agent that faithfully drafts whatever the research agent tells it to draft is behaving exactly as designed. If the research agent handed it something wrong, the drafting agent's behaviour will not trip any anomaly detector calibrated to that agent's own historical patterns.

"Agency isn't a feature. It's a transfer of decision rights.
The question shifts from is the model accurate to who is accountable when the system acts." [6]

In a multi-agent chain, that question of accountability does not have a single answer. It has to be answered at the level of the chain, which means the governance layer has to observe the chain as a unit, not assemble an answer after the fact from five separate, individually well-behaved logs.

The EU AI Act's high-risk obligations make this coordination problem more than an engineering concern. **Article 12 requires automatic logging throughout the operation of a high-risk AI system, while Article 14 requires human oversight measures that enable people to understand, monitor, interpret, and intervene in the system's operation where necessary.[10]** In a multi-agent architecture, neither obligation stops at the boundary of an individual agent.

The distinction between **logging and traceability** is critical. An organisation can log every LLM prompt and response and still be unable to reconstruct how a multi-agent decision was actually produced.

If Agent A's prompts and outputs sit in one logging table, Agent B's interactions sit in another, and Agent C's tool calls sit somewhere else, an auditor may have records of each individual interaction without having the **causal chain connecting them**.

Consider a decision that passes through five agents. Agent A retrieves information, Agent B interprets it, Agent C makes a recommendation, Agent D applies a compliance check, and Agent E executes the resulting action.

Static, siloed logs may show what each agent received and produced, but they do not necessarily establish **which instruction triggered which downstream action, what authority was delegated at each hop, what context was preserved or dropped, or where a critical decision changed.**

What is required for a multi-agent environment is therefore **execution graph traceability**: a linked record of the entire chain that allows an auditor or authorised human overseer to reconstruct the decision from its initiating instruction through every agent handoff, tool call, delegated permission, transformation, and final action.

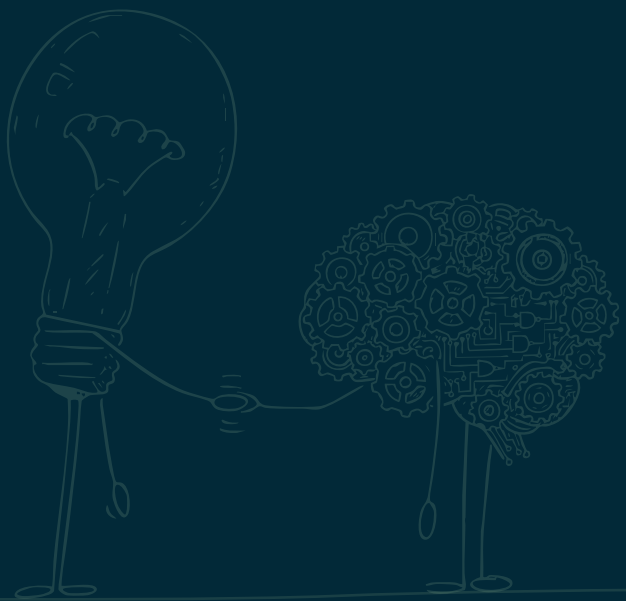
The execution graph must preserve the relationships between events, not simply store the events independently.

This also changes what meaningful human oversight looks like under **Article 14**. A human cannot effectively monitor or intervene in an autonomous system if the system presents only isolated agent outputs without showing how those outputs were generated or what upstream decisions and instructions shaped them.

Effective oversight requires sufficient visibility into the chain to identify where an error, unsafe instruction, lost constraint, or unauthorised delegation entered the workflow and to intervene before that failure propagates.

For multi-agent systems, compliance therefore cannot be demonstrated simply by producing a collection of prompt-and-response logs after an incident. **The organisation needs an execution record that makes the causal chain reconstructable end to end**.

Very few enterprises currently have this capability because their observability and governance infrastructure was designed around individual models and agents rather than the execution graph created when autonomous agents work together.



Section 4

Five Pillars of a Coordination-Ready Orchestration Layer

A coordination-ready orchestration layer is not a single tool, and it is not simply the sum of existing single-agent governance work. It is five interdependent capabilities purpose-built for the space between agents rather than the agents themselves.

Build chain-level discovery without delegated identity and the organisation can see the topology of its agent estate but still cannot prove who was authorised to do what at each hop.

Build handoff contracts without orchestration observability and context stops degrading in transit but nobody notices when the chain itself starts failing. The pillars have to be built together.

Pillar 1 - Chain-Level Discovery: Making the Agent Estate Visible

The first pillar extends agent discovery from an inventory of individual agents to a map of the relationships between them: which agents call which other agents, under what conditions, in what sequence, and how frequently, as a living topology rather than a one-off architecture diagram that goes stale the week it is drawn.

Modern orchestration frameworks generate this information as a natural by-product of how they route tasks; the gap is that almost nobody is capturing and centralising it.

Pillar 2 - Delegated Identity: Eliminating Token Over-Privilege Across Agent Hops

Rather than an agent carrying a single static identity through every call it makes, each handoff should establish a fresh, narrowly scoped delegation. Agent A tells Agent B to perform a specific task, and Agent B receives permissions explicitly scoped to that task rather than inheriting Agent A's broader access.

Short-lived, ephemeral tokens can enforce this boundary by limiting both the scope and lifetime of the delegated authority, reducing the risk that a credential remains usable beyond the task for which it was issued.

For stronger workload identity assurance, organisations can also use frameworks such as **SPIFFE/SPIRE** to attest the identity of each agent workload before issuing or accepting a delegation. This allows the orchestration layer to establish not only what authority is being delegated, but which verified workload is receiving it at each agent-to-agent hop.

The result is a chain of explicit, attributable delegations rather than a chain of inherited credentials. If Agent A delegates a task to Agent B, and Agent B subsequently invokes Agent C, the authority should be re-evaluated and re-scoped at each transition rather than passed through unchanged.

This limits privilege escalation and helps prevent confused-deputy scenarios in which a downstream agent inadvertently acts using permissions granted to an upstream agent for an entirely different purpose.

This is also what allows an audit trail to answer the question a regulator will actually ask: **not simply what did Agent B do, but who authorised it, which verified agent received that authority, what permissions were granted, for how long, and did the agent remain within the scope of the instruction?**

Pillar 3 - Structured Handoff Contracts: Preventing Context Loss Between Agents

Rather than agents passing each other free-text summaries of what they found or decided, a coordination-ready system should treat every agent-to-agent handoff as a **strict API contract**.

Schema-enforced JSON or Protobuf payloads can define exactly what information must be passed between agents, the permitted data types and values, and which fields are mandatory before the receiving agent can act.

This prevents critical context from being lost in prompt compression or natural-language summarisation. A handoff contract can require fields for the source of the information, confidence score, provenance, caveats, validation status, task scope, and any constraints that the receiving agent must preserve.

The receiving agent can then distinguish between information it is authorised to accept as validated input and information it must independently verify before taking action.

This changes the handoff from an unpredictable exchange of free-text prompts into a **machine-enforceable interface between autonomous agents**. If a required field is missing, malformed, outside its permitted range, or fails validation, the orchestration layer can reject or pause the handoff rather than allowing an incomplete context to propagate downstream.

Structured contracts also provide a consistent basis for **execution graph traceability.** Each payload can be linked to the originating agent, task, delegation, schema version, and downstream action, allowing the organisation to determine not only what information moved through the chain, but whether the information remained complete and within its defined contract at every hop.

The objective is not to eliminate natural-language reasoning inside agents. It is to ensure that **the boundary between agents is deterministic even when the reasoning inside each agent is probabilistic.**

This creates a controlled interface through which context, uncertainty, provenance, and constraints can travel reliably across the entire multi-agent workflow.

Pillar 4 - Orchestration Observability: Detecting Cascading Failure Before It Becomes an Incident

This is what most enterprises are missing entirely today, reflected in the finding that only 17% currently monitor agent-to-agent interactions continuously.[2]

Orchestration observability tracks the health of the chain as a whole: whether handoffs are completing within expected parameters, whether context is degrading measurably as it moves between agents, and whether a pattern of behaviour across multiple agents, none individually anomalous, is nonetheless inconsistent with how the chain is supposed to work.

Pillar 5- Deterministic Orchestration: Bounded Execution for High-Consequence Workflows

Not every workflow benefits from agents dynamically deciding, at runtime, which other agent to call next.

For workflows where the cost of an error is high or regulatory scrutiny is likely, a bounded, deterministic orchestration pattern, where the sequence of agents and the conditions for each handoff are explicitly defined and auditable in advance, is a meaningfully safer default than open-ended, agent-decided routing.

Section 5

What Coordination-Ready Orchestration Delivers

The case for investing in orchestration-layer governance is sometimes framed narrowly, as a way of avoiding the specific incident where a multi-agent chain does something visibly wrong. That framing understates the return.

Organisations that have built coordination-ready orchestration can scale multi-agent AI with a confidence that less prepared competitors simply cannot match, because they can see, prove, and predict how their agent estate behaves as a system.

When the coordination tax is invisible, cost and incident rates both tend to surprise the organisation at the worst possible moment. Chain-level discovery and orchestration observability turn that surprise into a measurable, forecastable curve.

Finance can see the true cost of adding a fourth agent to a chain before it is added, rather than discovering the impact on the next cloud bill.

This extends beyond predictable infrastructure costs to predictable compute and token spend. Unmonitored multi-agent systems can develop feedback loops in which agents repeatedly query, challenge, or re-invoke one another without reaching a terminal state.

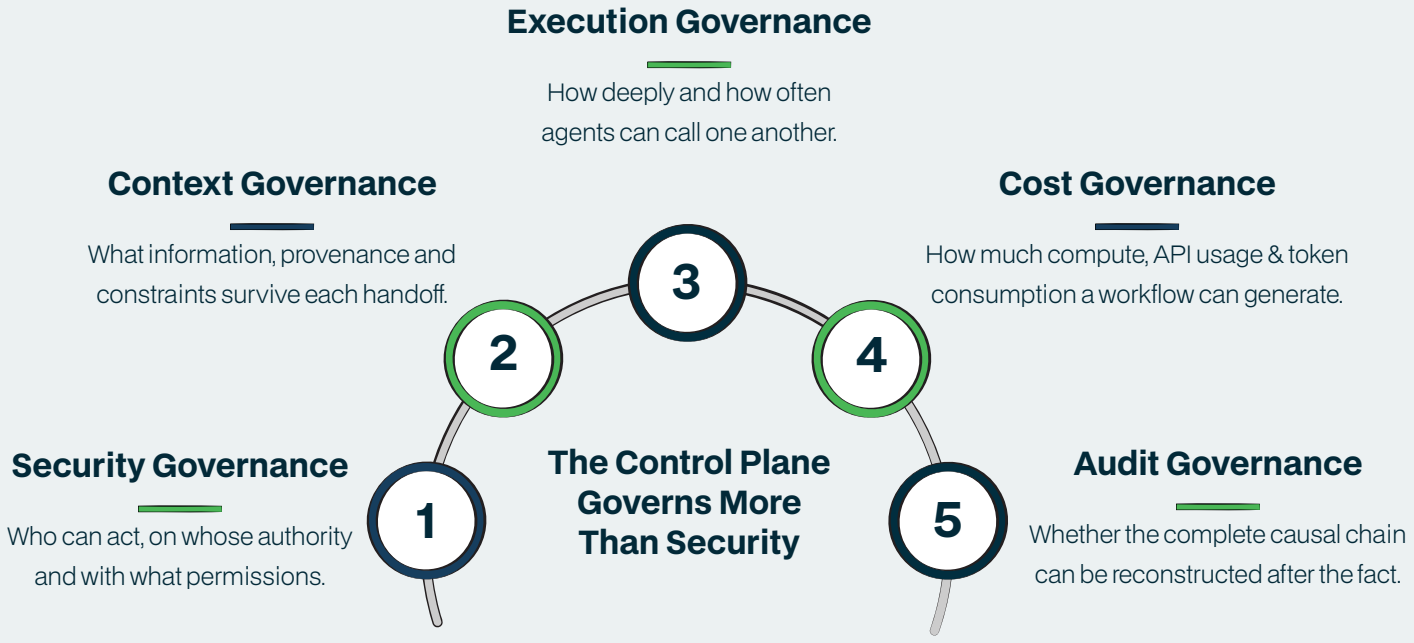
This infinite agentic reciprocation can occur when Agent A continually requests clarification from Agent B while Agent B responds by requesting additional information from Agent A, creating a loop that consumes model inference, API calls, compute and tokens without advancing the underlying task.

In a production environment, these loops can turn a coordination defect into a rapidly escalating cost event. The individual agents may each appear to be operating normally, while the orchestration layer is repeatedly triggering them.

Agent-level monitoring can therefore miss the commercial impact until the accumulated API and compute spend becomes visible.

A coordination-ready control plane should enforce cost governance alongside security governance. This means establishing controls such as maximum execution depth, hop limits, token and compute budgets, recursion detection, timeout policies, and circuit breakers that terminate anomalous agent-to-agent activity before it becomes an uncontrolled spend event.

The result is not simply safer orchestration. It is predictable orchestration economics: organisations can understand how many agents a workflow invokes, how many hops and tool calls it generates, how much context it processes, and what the expected compute and token cost is before that workflow scales into production.



The coordination layer therefore becomes the point at which enterprises govern both **what autonomous agents are allowed to do and how much resource they are allowed to consume while doing it.**

Section 6

A Framework for Closing the Gap

The most common mistake organisations make when they try to get ahead of the coordination tax is treating it as a bigger version of the single-agent governance work they have already started. It is related work, but it is not the same work. Better agent-level governance does not automatically add up to chain-level governance because the risk lives in the interactions between agents.

The practical approach is to establish a **Zero-Trust Control Plane for the existing agentic estate, rather than rebuild the estate** from scratch. Organisations do not need to abandon the frameworks and agents they have already invested in.

The priority is to gain visibility and control over how those agents interact, what authority moves between them, what context is exchanged, and what resources they consume.

Step 1 - Map the Chain Topology: Discover What Is Actually Running

Before any new tooling gets selected, the organisation needs an honest map of which agents call which other agents in production today, not the architecture diagram from the original design review, but what is actually running now.

This includes identifying conditional routes, parallel agent execution, tool calls, delegated identities, data dependencies, and potential recursion paths.

Step 2 - Classify AI Systems by Accountability Requirement

With the topology in hand, the priority shifts to instrumenting the handoffs themselves rather than only monitoring the agents at either end of them.

This means establishing delegated identity for the highest-risk chains, capturing execution-graph telemetry at every hop, monitoring context and authority propagation, and introducing controls for anomalous execution depth, recursion, compute and token consumption.

Step 3 - Standardise Handoff Contracts: Make Context Machine-Enforceable

Once the highest-risk chains are instrumented, the organisation can begin defining structured handoff contracts for the workflows that matter most.

Schema-enforced payloads should specify what fields, confidence levels, provenance, caveats, permissions, and constraints must travel between agents. The objective is to prevent an agent from receiving a technically valid response that has lost the context required to interpret it safely.

Step 4 - Embed Continuous Orchestration Governance: Make Controls Inherit by Default

New chains should inherit the topology mapping, delegated identity, handoff contract, observability, and cost-governance patterns already established for existing workflows, rather than each new agent addition starting the risk conversation from zero.

Orchestration observability then runs continuously, detecting cascading failures, privilege anomalies, context degradation, runaway loops, and unexpected resource consumption before they become customer-facing, security, compliance, or cost incidents.

Merit's Approach: Enterprise Agentic Control Plane Integration

Merit's approach is designed around Enterprise Agentic Control Plane Integration, rather than rip-and-replace transformation. Organisations that have already built multi-agent workflows using CrewAI, AutoGen, LangGraph, or other agentic frameworks do not need to discard those investments.

Merit's KIAA framework can be deployed as an orchestration and observability control-plane wrapper around existing multi-agent chains, providing the governance capabilities that are typically missing between individual agents.

Rather than replacing the underlying agents, KIAA provides the layer through which their interactions can be discovered, governed, observed, traced, and controlled.

This allows enterprises to bring existing multi-agent workflows under a common control model, with capabilities spanning chain-level topology, delegated identity, structured handoffs, execution-graph traceability, orchestration observability, and compute and token governance. Existing agents can continue to perform their specialised functions while the control plane governs the interaction between them.

THE ARCHITECTURAL PRINCIPLE

Keep the agents that already work.

Add the control plane they are missing.

This approach also reduces the friction associated with enterprise adoption.

Instead of undertaking a wholesale migration from an existing CrewAI, AutoGen, or LangGraph implementation, organisations can introduce coordination controls around their current estate and progressively bring higher-risk workflows under governance.

The result is a multi-agent environment that is more auditable, more controllable, and **more predictable without requiring the underlying agentic architecture to be rebuilt from the ground up**.

The objective is to make existing autonomous workflows governable at the point where the greatest risk emerges: **the space between the agents.**

Conclusion

The question facing every CDO, CISO, and CTO running agentic AI in 2026 is no longer whether to chain agents together. That decision has already been made, workflow by workflow, across most large enterprises, often without anyone deciding it deliberately.

The question is whether the organisation can see the chains it has built, govern the handoffs between them, and explain what happened when something goes wrong at a seam rather than inside any single agent.

The coordination tax does not announce itself. It accumulates quietly, one sensible agent addition at a time, until a chain that nobody designed as a system produces a failure that nobody can fully explain.

Organisations that get ahead of this in 2026 will scale multi-agent AI with a confidence their competitors cannot match. The ones that do not will spend 2027 reconstructing chains after the fact, under regulatory and reputational pressure, trying to answer a question their architecture was never built to answer.

2026 is not the moment to write a policy about how agents should coordinate. It is the moment to find out how they actually are, and start building the layer that governs the space between them.

Ready to Understand Your Agent Estate?

Your organisation may already have more agent-to-agent relationships than its governance programme can see.

Merit's AI and data engineering teams can help you establish what is actually running across your agent estate and identify where coordination risk is accumulating.

The Agent Estate Topology & Handoff Risk Audit examines which agents are operating across your estate, how they call and interact with one another, where delegated authority and credentials move between agents, where context, confidence and provenance can be lost, where handoff contracts are missing, and where execution-graph traceability is incomplete.

It also identifies where recursive or runaway execution could occur, where compute and token consumption may become unpredictable, and which workflows present the highest coordination and compliance risk.

From there, we can define a practical path towards an **Enterprise Agentic Control Plane**, including chain topology discovery, delegated identity, structured handoff contracts, execution-graph observability and cost governance.

You do not need to replace the agents you have already built.

Start by understanding how they are actually working together.

Book an Agent Estate Topology & Handoff Risk Audit with Merit.

meritdata-tech.com/ai

About Merit Data & Technology

Merit Data & Technology, part of Merit Group PLC, is a trusted partner in AI-driven data and digital transformation. With over two decades of experience, we deliver practical, ethical, and scalable AI solutions for complex enterprise challenges. Our AI services span AI-driven automation, business process intelligence, the KIAA modular AI framework, text and image analytics, and AI-powered data extraction and classification. We work side-by-side with clients to design multi-agent AI systems that perform today and are built to be coordinated, monitored, and explained as they scale.

Sources

1. Salesforce 2026 Connectivity Benchmark Report. The report published that the average company runs 12 AI agents today, expected to reach 20 by 2027, but 50% of those agents operate completely on their own. salesforce.com/in/news/stories/connectivity-report-announcement-2026/

2. EY / AIUC-1 Consortium (2026). Survey published in Help Net Security, March 2026. Finding: only 38% of organisations monitor AI traffic end-to-end across prompts, tool calls, and outputs; 17% continuously monitor agent-to-agent interactions. helpnetsecurity.com

3. Gartner (2025). Press Release: Gartner Predicts Over 40% of Agentic AI Projects Will Be Canceled by End of 2027. June 25, 2025. gartner.com/en/newsroom/press-releases/2025-06-25-gartner-predicts-over-40-percent-of-agentic-ai-projects-will-be-canceled-by-end-of-2027

4. Bain & Company (2026). The Three Layers of an Agentic AI Platform. Analysis of orchestration, memory management, and governance as first-class infrastructure concerns. bain.com/insights/the-three-layers-of-an-agentic-ai-platform/

5. Google Cloud (2025). Google Cloud Cybersecurity Forecast 2026. Analysis of agentic identity management and chained-agent risk for enterprises. cloud.google.com/security/resources/cybersecurity-forecast

6. McKinsey & Company (2026). State of AI Trust in 2026: Shifting to the Agentic Era. AI Trust Maturity Survey of approximately 500 organisations. mckinsey.com/capabilities/tech-and-ai/our-insights/tech-forward/state-of-ai-trust-in-2026-shifting-to-the-agentic-era

7. McKinsey & Company (2026). Trust in the Age of Agents: Agentic AI Governance for Autonomous Systems. mckinsey.com/capabilities/risk-and-resilience/our-insights/trust-in-the-age-of-agents

8. Cloud Security Alliance (2026). Research Note: AI Agent Governance Framework Gap. AI Controls Matrix (AICM) covering 18 security domains and over 240 control objectives mapped to the AI lifecycle. cloudsecurityalliance.org/research

9. Deloitte AI Institute (2026). State of AI in the Enterprise 2026. Survey of 3,235 senior leaders across 24 countries. deloitte.com/us/en/what-we-do/capabilities/applied-artificial-intelligence/content/state-of-ai-in-the-enterprise.html

10. European Union (2024). Regulation (EU) 2024/1689 - Artificial Intelligence Act. Article 12 (automatic logging) and high-risk AI obligations entering full force August 2026. eur-lex.europa.eu/eli/reg/2024/1689/oj

11. Anthropic (2025). Model Context Protocol Specification. Technical documentation on agent-to-agent and agent-to-tool connectivity patterns. modelcontextprotocol.io

12. Microsoft (2026). AutoGen and Multi-Agent Orchestration: Design Patterns for Enterprise Deployment. Analysis of agent-to-agent delegation and coordination patterns. microsoft.com/research

13. MIT Sloan Management Review (2026). Coordinating the Enterprise: The Governance Gap in Multi-Agent AI. Analysis of cascading failure modes in chained autonomous systems. sloanreview.mit.edu

14. Bain & Company (2026). Building the Foundation for Agentic AI: Technology Report. Analysis of legacy batch-based system limitations and the agentic thundering herd effect on data infrastructure. bain.com/insights/building-the-foundation-for-agentic-ai-technology-report-2025/

15. Gartner (2025). Forecast cited in press release: 40% of enterprise applications will embed task-specific AI agents by end of 2026, up from less than 5% in 2025. gartner.com/en/newsroom/press-releases/2025-06-25-gartner-predicts-over-40-percent-of-agentic-ai-projects-will-be-canceled-by-end-of-2027