CrAIg™

Version 1.0
June 2026
CONTINUOUS RUNTIME AI GOVERNANCE

Cross-System Constraint Collisions: The Governance Gap in Enterprise Agentic AI

A technical and operational framework for cross-system constraint governance in autonomous agentic environments

Michael Mallon
Founder, HimalAIan, LLC™
michael.mallon@himalaian.com

Patent Pending: HIMA101PR (App. No. 64/067,005) and HIMA102PR (App. No. 64/076,093)

01 Executive Summary

Enterprises have invested heavily in AI agents, enterprise resource planning (ERP) systems, integration middleware, and workflow automation. Each platform governs itself. None governs the space between them.

A new category of operational risk has emerged at the intersection of enterprise AI and multi-system workflow execution. As autonomous artificial intelligence (AI) agents increasingly coordinate decisions across independently operated platforms - inventory systems, customer relationship management (CRM) systems, financial systems, compliance platforms, and integration middleware - the potential for cross-system contradictions has grown substantially.

The risk is not that individual systems malfunction. Each platform, operating independently, may function exactly as designed. The risk is that the interaction of correct signals from multiple systems produces an incorrect aggregate outcome: one that no single system is positioned to detect, and that no governance framework reviewed for this paper is designed to prevent.

This paper defines that risk formally, examines how it emerges in practice, and introduces an architectural framework for governing it: continuous runtime governance of autonomous agent workflows across the full transaction path, not just within individual platform boundaries.

The framework introduced here - CrAIg™ (Continuous Runtime AI Governance) -- is not presented as a completed product. It is presented as an architectural response to a documented and growing enterprise governance gap. The goal of this paper is to define that gap precisely enough that enterprise architects, operations leaders, and technology investors can evaluate its significance independently of any commercial claim.

02 The Enterprise Shift: AI Agents in Operational Workflows

For most of the past decade, artificial intelligence in enterprise environments operated as a recommendation layer. AI systems analyzed data and produced insights. Humans decided what to do with those insights. The governance question was relatively contained: was the recommendation accurate, and was it appropriately disclosed?

That model is changing. AI agents in 2025 and 2026 are not recommendation engines. They are workflow participants. They initiate actions, coordinate across systems, and execute decisions with real operational consequences - often without a human in the loop at each step.

Where AI agents now operate

Autonomous AI agents currently influence or execute decisions across the following enterprise domains:

- Supply chain and inventory allocation: determining which products are assigned to which commitments, in what sequence, under what constraints
- Customer commitment management: generating, modifying, and communicating order confirmations, substitution proposals, and delivery windows
- Financial exposure management: evaluating credit limits, margin thresholds, and contractual obligation exposure in real time
- Regulatory compliance coordination: verifying product certifications, import documentation, and safety holds before fulfillment actions execute
- Integration middleware routing: transforming, enriching, and routing transaction data between enterprise platforms via orchestration layers such as Celigo, MuleSoft, and Boomi
- ERP workflow coordination: triggering downstream actions in NetSuite, SAP, and equivalent systems based on upstream signals from CRM and commerce platforms

The operational scope of AI agent action has expanded from advisory to executive. That expansion creates a governance requirement that existing enterprise architecture was not designed to satisfy.

The speed asymmetry problem

AI agents execute at machine speed. Governance frameworks, where they exist, were designed for human-speed decision cycles. An agent coordinating a fulfillment allocation across five enterprise systems can traverse multiple workflow steps, touch multiple customer commitments, and initiate multiple downstream actions in the time it takes a human reviewer to open an email notification.

This speed asymmetry is not inherently problematic when each step is governed correctly within its own system boundary. It becomes operationally dangerous when the correct execution of multiple steps - each individually valid - produces a cross-system contradiction that no single system is designed to detect.

> *The problem is not that AI agents move too fast for humans to review every action. The problem is that no system is positioned to evaluate the aggregate context of all relevant connected systems at the moment each action executes.*

03 The Governance Gap: What Existing Systems Cannot See

Enterprise governance frameworks - ERP approval workflows, compliance monitoring systems, AI observability platforms, and integration middleware - share a common architectural assumption: governance is evaluated within the boundary of a single system or a single agent.

That assumption was reasonable when enterprise systems were largely independent and human-coordinated. It is no longer sufficient when autonomous AI agents execute multi-step workflows across multiple independently operated platforms simultaneously.

Three documented failure modes

1. The single-platform governance limitation

The governance frameworks reviewed for this paper - including Microsoft's Agent Governance Toolkit, [2] Oracle's Runtime Governance for Enterprise Agentic AI, [3] the MI9 framework (MI9 is a framework designation, not a spelled-out acronym), [1] and ServiceNow's AI Control Tower - govern agents within a defined scope: a single platform, a single fleet, or agents connected through a proprietary orchestration layer. No reviewed framework was found to address constraint violations that emerge from the interaction of proposed actions across multiple independently operated platform boundaries simultaneously.

ServiceNow's AI Control Tower merits specific examination. At Knowledge 2026 in May 2026, ServiceNow positioned AI Control Tower as governance infrastructure spanning AWS, Microsoft Azure, Google Cloud, SAP, Oracle, Workday, and 25 additional enterprise systems - capable of discovering, observing, governing, securing, and measuring AI agents "regardless of where they run." [10] ServiceNow's governance model is cross-platform control, identity, observability, and workflow governance: it controls which agents are authorized to act, applies access policies through its Veza-powered identity graph, and can issue kill switches when agents exceed permitted boundaries. CrAIg™ is cross-platform constraint collision detection and verdict resolution: it evaluates whether the constraint governing an agent's next proposed action conflicts with a constraint sourced from another independently operated platform, and determines which governs when they conflict. Those are structurally different problems, and ServiceNow's architecture does not address the second.

A governance system that monitors only the inventory platform cannot detect a collision between a proposed inventory action and a financial exposure constraint. A governance system that monitors only the CRM cannot detect a collision between a proposed customer priority action and a compliance hold. The constraint violation exists in the interaction of proposed actions against cross-system context - not in any individual system.

2. The inter-step visibility gap

When an autonomous AI agent executes a multi-step workflow across multiple enterprise systems, the context of each connected system may change between workflow steps. A compliance hold may be applied to a product lot after the inventory system has confirmed availability. A customer's credit limit may be exceeded after the CRM has ranked that customer as a priority account. A middleware sync may partially fail after a customer-facing system has confirmed order fulfillment.

These changes are invisible to the agent and to any single-platform governance layer. The agent operates on the context it observed at the beginning of the workflow. No reviewed

governance mechanism re-evaluates cross-system constraint context at each step boundary to detect changes that have occurred since the workflow began.

3. The irreversibility risk

Autonomous agents executing multi-step workflows across disconnected systems routinely take irreversible actions: customer notifications are sent, inventory is committed, financial reservations are made, regulatory filings are initiated. Each of these actions may be individually appropriate at the moment it executes. Collectively, they may commit the organization to an outcome that a cross-system governance review would have prevented.

By the time the cross-system contradiction becomes visible - typically when a human reviews a dashboard, a report, or a customer complaint - multiple irreversible actions have already been taken across multiple systems. The cost of resolution is substantially higher than the cost of prevention.

Existing platform governance	Cross-system governance (CrAIg™)
Evaluates proposed actions within one system boundary	Evaluates proposed actions using cross-system context across all connected systems
Detects constraint violations at the platform layer	Detects constraint violations that emerge between platforms
Governs before or after workflow execution	Governs at each execution step boundary, before action executes
Reviews actions after they execute	Halts actions before they execute when a collision is detected
Produces platform-specific audit records	Produces a unified cross-system audit trail

04 The Transaction Path Problem

> *The most important architectural insight in enterprise AI governance is not about the destination system. It is about the transaction path.*

Enterprise integration architecture has historically focused governance attention on systems of record: specifically the ERP, the compliance database, and the financial ledger. These are the authoritative sources of truth for what has happened. They are, by definition, retrospective.

AI-driven enterprise workflows introduce a new governance surface: the transaction path. The path between a triggering event (a purchase order, a harvest report, a fulfillment request) and its committed outcome in the system of record passes through multiple independently operated systems, each making decisions based on incomplete information about the others' current context.

The three-layer transaction topology

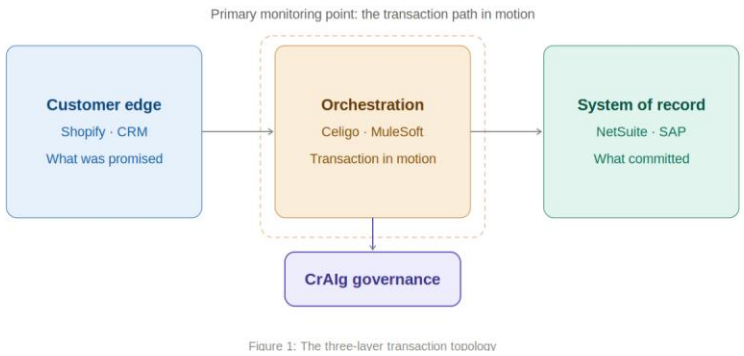

Figure 1: The three-layer transaction topology

Customer edge layer

The customer-facing systems that capture commitments and communicate outcomes: Shopify, Salesforce Commerce, and equivalent platforms. This layer records what the customer was promised - order confirmation, fulfillment status, delivery window. It is the most dangerous layer to have out of sync with the system of record.

Orchestration layer

The integration middleware that transforms, routes, enriches, and synchronizes data between systems: Celigo, MuleSoft, Boomi, and equivalent platforms. This layer is the transaction path itself - where data moves between systems, where transformations occur, where failures propagate, and where timing gaps emerge. Critically, the orchestration layer sees the transaction in motion before it commits anywhere.

System of record layer

The authoritative enterprise platforms that commit outcomes: NetSuite, SAP (Systems, Applications, and Products), and equivalent ERP systems, along with compliance and financial systems. This layer records what actually happened -- what inventory was allocated, what revenue was recognized, what compliance status was applied. It is retrospective by nature.

> *Governing only the system of record is governing the outcome after it has committed. Governing the orchestration layer is governing the transaction while it is still in motion. The highest-value governance point in an AI-driven enterprise workflow is the orchestration layer - because that is where cross-system contradictions first become visible, and where intervention can still prevent irreversible action.*

This architectural insight - that governance must follow the transaction path, not just the destination system - is the foundational principle of the framework described in this paper.

05 Constraint Collisions: A Taxonomy

A constraint collision is a condition in which the cross-system context of two or more independently operated enterprise systems, evaluated against a proposed agent action, reveals a violation of at least one operational, financial, regulatory, or relational constraint - where that violation is undetectable by any single system operating independently.

Constraint collisions are distinct from system errors. Each system involved may be functioning correctly. The collision emerges from the interaction of correct signals evaluated against a proposed action, not from any individual system's failure.

Formal Definition	A constraint collision exists when: (1) an agent proposes an action; (2) at least two independently operated systems each hold context relevant to that action; (3) evaluating the proposed action against the combined cross-system context reveals a constraint violation; and (4) no single system, operating independently, holds sufficient context to detect the violation. All four conditions must be present. A violation detectable by a single system is a platform-level enforcement failure, not a constraint collision.

> ***A violation detectable by a single system is a platform-level enforcement failure, not a constraint collision.***

Four documented collision types

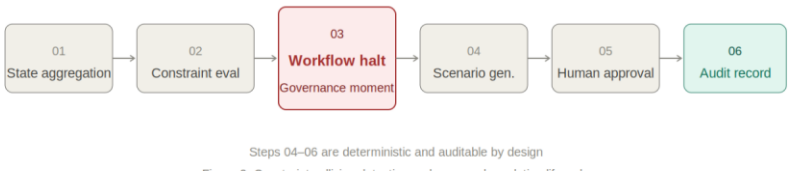

Figure 2: Constraint collision detection and governed resolution lifecycle

State drift

The context of one system changes after a connected system has acted on its prior context. A product lot is placed on regulatory hold after the inventory system has confirmed its availability to the allocation agent. A customer's credit balance changes after the CRM has ranked that customer for priority fulfillment. The agent continues operating on prior context - because no mechanism exists to re-evaluate cross-system constraint context between steps.

Timing divergence

Two systems make correct decisions at different points in time, producing contradictory committed outcomes. A commerce platform confirms an order and dispatches fulfillment notifications. An integration middleware layer partially fails in transmitting that order to the ERP. The ERP creates a backorder rather than a committed allocation. The commerce platform's state (fulfilled) and the ERP's state (backorder) diverge - and no system is positioned to detect the contradiction.

Partial middleware failure

An orchestration flow partially succeeds - completing some transformation and routing steps while failing others. The partial success is sufficient to commit state in the destination system but insufficient to reflect that state correctly in the originating system. The result is a cross-system contradiction that is invisible to both systems and to any single-platform governance layer.

Contradictory workflow truths

Two systems independently derive contradictory conclusions about the same entity from their respective data sets. A CRM ranks a customer as highest priority for allocation based on relationship history. A financial system identifies that same customer as over their credit limit based on outstanding receivables. Both conclusions are correct within their respective data contexts. The contradiction exists only in the interaction - and only a governance layer with simultaneous cross-system context visibility can detect it when the agent proposes an action affecting that customer.

> *In each case, the collision is only visible to a governance layer that simultaneously evaluates the proposed action against cross-system context. No reviewed platform or governance framework was found to address these specific four-condition constraint collision scenarios at runtime.*

06 Workflow Halt and Escalation: Governed Interruption

The appropriate response to a detected cross-system constraint collision is not automatic remediation. It is governed interruption: halting the workflow before any irreversible action executes, generating a structured set of resolution options, and routing the decision to a human with the authority, context, and information required to make it correctly.

This principle - that governance does not replace human judgment but informs it - is architecturally important. The value of a runtime governance system is not that it makes better decisions than humans. It is that it ensures humans are presented with the right decision at the right moment, with the right information, before the cost of that decision becomes irreversible.

The governed interruption sequence

- Detection: the CECR (Constraint Evaluation and Collision Resolution) engine identifies a constraint violation by evaluating the proposed action against cross-system context at each workflow step boundary
- Halt: the Workflow Halt and Escalation Gate prevents execution of the pending workflow action before any irreversible downstream action is initiated
- Scenario generation: the Probabilistic Financial Scenario Engine generates a structured set of resolution paths, each evaluated for financial impact, customer risk, compliance status, and operational feasibility
- Escalation: the Governed Recommendation Surface routes the ranked scenarios to the designated human decision-maker, with all relevant context and expected value modeling
- Human approval: the decision-maker selects a resolution path, documents the rationale, and authorizes resumption of governed workflow execution
- Audit: the Cross-System Audit Trail records the complete governance event - collision detected, workflow halted, scenarios generated, decision made, execution resumed

The dismissal with intent principle

A governed interruption system must account for the operational reality that human decision-makers are not always immediately available, and that some detected collisions may be consciously deferred rather than immediately resolved.

The appropriate design response is not to make deferral impossible - that would create unacceptable workflow friction. It is to make deferral conscious and auditable. A decision-maker who chooses to defer resolution of a detected collision should be required to record a brief reason for that deferral. That record creates an irrefutable audit artifact: the decision-maker was presented with the collision, understood its nature, and chose to defer resolution for a documented reason.

This principle - dismissal with intent - satisfies the governance requirement without imposing unacceptable operational friction. It also produces a materially stronger audit record than a system that either prevents deferral or permits it silently.

> *"I never saw the alert" is not a defensible governance position when the audit trail shows the decision-maker typed a reason into a dismissal field.*

07 Runtime Governance Architecture

The following describes the architectural components of a cross-system runtime governance framework. Each component addresses a specific gap in existing enterprise governance architecture. Together they form a governance layer that operates between autonomous agent execution steps, across the full transaction path.

The CECR seven-stage evaluation trace

Every governance decision produced by CrAIg™ passes through a deterministic seven-stage evaluation trace within the CECR engine. The engine evaluates each proposed agent action using cross-system context - which may include event signals, constraint metadata, and, where configured, state-derived inputs from connected platforms. This trace is the architectural centerpiece of the framework and produces an immutable audit record for every decision:

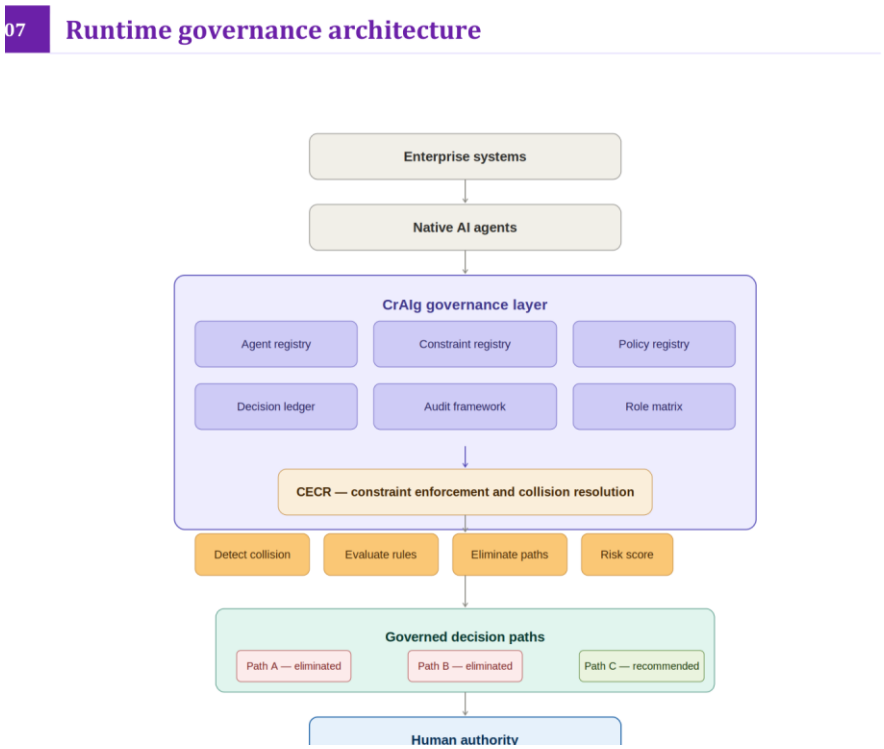

Figure 3: CrAIg™ governance architecture

```
event_received → rules_loaded → rules_evaluated → rules_matched →
collision_checked → verdict_issued → audit_written
```

Rule precedence within the CECR engine is absolute: Block overrides Review, Review overrides Pass. The most restrictive verdict wins. There is no silent override path; all exception handling is explicit, authorized, and audited. This deterministic rule architecture is what makes CrAIg™'s audit trail legally defensible and regulatory-compliant.

Architectural components

A	**Cross-System Context Monitor** Maintains a runtime cross-system context model aggregating signals from connected enterprise platforms, updated via event-driven connectors and configurable polling intervals. Evaluates constraint implications across system boundaries at each workflow step by processing event signals, constraint metadata, and, where configured, state-derived inputs - producing a composite view of cross-system context that no individual platform can generate independently. Connected via Model Context Protocol (MCP) connectors, Representational State Transfer (REST) API integrations, webhook subscriptions, event stream subscriptions, and enterprise service bus integrations.
B	**CECR: Constraint Evaluation and Collision Resolution Engine** The architectural centerpiece of CrAIg™. At each execution step boundary, evaluates each proposed agent action against the active constraint rule set using cross-system context through the seven-stage evaluation trace. Detects constraint violations that emerge from the interaction of proposed actions against multi-platform context - violations undetectable by any single connected system. Evaluates both hard-stop constraints (unconditional workflow halt) and soft constraints (scenario generation and human review). Rule precedence is deterministic: Block overrides Review, Review overrides Pass, most restrictive verdict wins.
C	**Workflow Halt and Escalation Gate** Upon collision detection, prevents execution of the pending workflow action and suspends autonomous agent execution before any irreversible action is taken. Configurable to apply graduated escalation responses: full workflow suspension, partial suspension of the pending action, automatic rollback of reversible prior actions, or escalation notification with a governed resolution deadline.
D	**Probabilistic Financial Scenario Engine** Generates a structured set of governed resolution scenarios following collision detection. Employs a two-layer evaluation architecture: a deterministic rules engine evaluates constraint compliance for each candidate resolution path; a large language model (LLM)-assisted reasoning layer models financial impact, customer risk, and operational feasibility across multiple variable dimensions. Governance is always deterministic. Intelligence is always augmented.
E	**Governed Recommendation Surface** Presents ranked resolution scenarios with expected value modeling to the designated human decision-maker for review and approval. Does not autonomously select or implement a resolution scenario without explicit human authorization. Routes decisions to the appropriate role based on financial exposure threshold and organizational approval configuration, with support for backup approvers and escalation pathways.
F	**Cross-System Audit Trail** Records a complete, timestamped governance log spanning all connected platform systems at each workflow step: constraint collision events, workflow suspensions, resolution scenarios generated, human approval decisions, and resumed workflow actions. The seven-stage CECR evaluation trace is written to the audit record for every governance decision. Structured to satisfy regulatory traceability requirements including FDA FSMA 204 Key Data Element documentation and EU AI Act Article 12 tamper-evident audit trail obligations.

Constraint rule architecture

The constraint rule set evaluated by the CECR engine is organized in a hierarchical structure:

- Hard-stop constraints: violations unconditionally trigger workflow halt regardless of financial or relationship implications. Includes regulatory compliance requirements, product safety holds, financial exposure limits beyond configured thresholds, and data integrity violations including documentation discrepancies and chain-of-custody failures.
- Soft constraints: violations trigger scenario generation and human review but do not unconditionally require workflow halt. Includes relationship risk thresholds, margin degradation parameters, substitution eligibility boundaries, and configurable business rule parameters.

The configurable constraint rule set architecture accepts domain-specific rule sets and probabilistic model variables as inputs, enabling deployment across multiple industry verticals without modification to the core governance engine. This vertical-agnostic architecture is a deliberate design decision: the governance problem is structural, not industry-specific.

08 Vertical Examples

The following examples illustrate how cross-system constraint collisions manifest in specific industry contexts. In each case, the collision is structural: it emerges from the evaluation of a proposed action against cross-system context held across independently operated systems, and is undetectable without simultaneous cross-system context visibility.

Perishable goods distribution

A seafood distribution operation maintains pre-season volume commitments to wholesale and foodservice customers. An autonomous AI agent coordinates harvest allocation across five independently operated systems: inventory, sales, CRM, financial, and regulatory compliance.

When confirmed harvest volumes fall short of aggregate commitments, the allocation agent begins executing a reallocation workflow. Cross-system constraint collisions that emerge during execution include:

- An FDA import alert fires on a specific product lot in the compliance system after the inventory system has confirmed that lot's availability and the sales system has begun generating customer allocation notifications. The collision - between a proposed allocation action, confirmed available inventory context, pending notification context, and compliance hold context - is visible only through simultaneous cross-system evaluation.
- A customer's outstanding receivables balance exceeds the credit limit threshold in the financial system after the CRM has ranked that customer as the highest-priority allocation account. Two systems hold contradictory context about the same customer; the collision becomes visible only when a proposed allocation action is evaluated against both.
- A Country of Origin Labeling (COOL) documentation discrepancy is identified in the compliance system for the proposed substitute species after the inventory system has confirmed availability and the sales system has generated draft substitution proposals.

Each collision is detectable only through simultaneous evaluation of proposed actions against cross-system context. Each represents a point at which uninterrupted autonomous execution would produce an outcome with material financial, regulatory, or relational consequences.

eCommerce fulfillment

A retail operation uses Shopify as its customer-facing commerce platform, Celigo as its integration middleware, and NetSuite as its ERP system of record. An autonomous AI agent coordinates order processing across this three-layer stack.

A batch of 847 orders is received by Shopify. Payment is captured and fulfillment confirmation emails are dispatched. Celigo initiates the synchronization flow to NetSuite. During enrichment, a field mapping mismatch causes Celigo to submit a partial payload to NetSuite. NetSuite receives the partial data and creates backorders rather than committed allocations. Shopify's operational context: fulfilled. Celigo's operational context: partial failure, retrying. NetSuite's operational context: backorder.

Three systems, three versions of reality, for the same 847 transactions. Customer service, warehouse, and finance teams are operating from different truths. No single system is positioned to detect the contradiction - because each system's individual state is internally consistent.

Manufacturing and supply chain

A powersports manufacturer operates AI agents across production planning, procurement, compliance, and logistics systems. Cross-system constraint collisions in this environment include per- and polyfluoroalkyl substances (PFAS) regulatory classification changes that conflict with in-progress procurement orders, or supplier qualification status changes that affect committed production schedules mid-workflow.

Healthcare and MedTech

AI agents coordinating patient scheduling, insurance authorization, clinical protocol compliance, and supply chain for disposable medical devices face constraint collisions when insurance authorization context changes after scheduling confirmations have been issued, or when regulatory classification changes affect device inventory commitments mid-workflow.

09 Why Existing Platforms Do Not Solve This

The governance gap described in this paper is not fully addressed by any reviewed enterprise software category. Understanding why requires examining what each category was designed to do -- and what it was not.

ServiceNow AI Control Tower

ServiceNow is the most prominent vendor claiming cross-platform AI governance as of May 2026. At Knowledge 2026, ServiceNow positioned AI Control Tower as governance infrastructure spanning AWS, Microsoft Azure, Google Cloud, SAP, Oracle, Workday, and 25 additional enterprise systems - capable of discovering, observing, governing, securing, and measuring AI agents "regardless of where they run." [10] The platform acquired Traceloop for runtime observability, Veza for identity-based access graph technology, and Armis for asset intelligence across IT, OT, and IoT environments.

ServiceNow's governance model is cross-platform control, identity, observability, and workflow governance: it controls which agents are authorized to act, applies access policies derived from its Veza-powered identity graph, and can issue kill switches when agents exceed permitted boundaries. This is a significant and well-resourced governance capability, and a serious adjacent platform.

What it does not address is the constraint collision problem as defined in Section 05. ServiceNow governs which agents are authorized to act and through which channels. It does not evaluate whether a proposed agent action - governed by a constraint from Platform A - conflicts with a constraint from Platform B operating on the same transaction simultaneously. ServiceNow is cross-platform control and workflow governance. CrAIg™ is cross-platform constraint collision detection and verdict resolution. These are complementary, not competing, governance layers.

ERP and workflow management systems

ERP platforms govern workflow execution within their own data model and transaction boundaries. NetSuite, SAP, and equivalent systems enforce approval workflows, financial controls, and audit logging for transactions that originate and commit within the ERP. They cannot evaluate constraint implications that require simultaneous visibility into the context of independently operated external systems.

AI observability and monitoring platforms

AI observability platforms track model performance, inference latency, and output quality for AI systems. They are instrumentation layers, not governance layers. They observe what AI systems do; they do not evaluate whether proposed agent actions are permissible given the cross-system constraint context.

Integration middleware

Middleware platforms including Celigo, MuleSoft, and Boomi orchestrate data flow between systems. They are the orchestration layer - the transaction path itself. They are not governance layers. They execute routing and transformation logic; they do not evaluate whether the aggregate outcome of that routing and transformation is consistent with cross-system constraint requirements. Middleware failure is, in fact, one of the primary sources of the constraint collisions described in this paper.

Robotic process automation (RPA)

RPA platforms automate rule-based workflow steps within defined boundaries. They are not designed for multi-step autonomous workflows across heterogeneous systems, and they do not include cross-system context aggregation or constraint collision detection capabilities.

AI copilots and assistant frameworks

AI copilot frameworks assist human decision-makers with analysis and recommendation generation. They operate in advisory mode. They do not govern autonomous agent execution, and they do not intercept workflow actions before they execute.

> *Enterprises already have AI. They already have ERPs, middleware, and monitoring tools. What they do not have is a governance layer that evaluates proposed agent actions against cross-system constraint context simultaneously - governing the transaction path, not just the destination.*

10 Future Direction: A New Governance Layer for Enterprise AI

The governance gap described in this paper is not a transitional problem that will resolve as AI systems mature. It is a structural consequence of the multi-system architecture of enterprise operations - an architecture that will become more complex, not less, as AI agents take on broader workflow responsibilities.

As agentic AI systems extend their operational scope, the number of independently operated systems they coordinate across will increase. The number of potential cross-system constraint collisions will grow proportionally. The speed at which those collisions can produce irreversible consequences will accelerate.

The emerging governance category

What is described in this paper is the foundational architecture of a new enterprise software category: cross-system constraint governance middleware. This category sits at the intersection of three established disciplines - AI governance, enterprise integration, and operational risk management - and addresses a problem that none of the three disciplines currently solves in isolation.

The characteristics of this category are:

- Vendor-neutral: the governance layer connects to existing enterprise platforms via standardized protocol connectors, without requiring modification to those platforms or replacement of existing infrastructure
- Vertical-configurable: the constraint rule set is domain-specific; the governance engine is domain-agnostic, enabling deployment across multiple industry verticals without architectural modification
- Human-in-the-loop by design: the framework does not replace human judgment; it ensures that human judgment is applied at the right moment, with the right information, before the cost of a decision becomes irreversible
- Audit-native: the Cross-System Audit Trail is not a secondary feature; it is a primary output of the governance architecture, designed from the ground up to satisfy emerging regulatory traceability requirements

Regulatory tailwinds

Two regulatory frameworks create structural demand for cross-system audit trails of autonomous AI action:

- EU AI Act Article 12: tamper-evident logging requirements for high-risk AI systems. The August 2026 enforcement deadline creates immediate compliance urgency for enterprise operators.[5] CrAIg™'s CECR seven-stage evaluation trace is designed to support compliance with Article 12's authorization decision logging requirements - producing a timestamped, immutable record of every constraint evaluation and governance verdict.
- FDA Food Safety Modernization Act Section 204 (FSMA 204), compliance deadline July 20, 2028: mandates electronic sortable records of Key Data Elements for Critical Tracking Events, within a 24-hour response window.[4] For food distributors deploying autonomous AI agents across supply chain systems, this requirement creates structural demand for the kind of cross-system audit trail described in this paper.

These regulatory frameworks do not merely create demand for documentation. They create demand for governance architectures that make that documentation possible - architectures that maintain cross-system visibility at each workflow step boundary and record governance events in a structured, queryable, audit-ready format.

> *The governance gap described in this paper will narrow over time as enterprise architecture evolves. The organizations that define and implement cross-system constraint governance frameworks now will be positioned as the infrastructure layer for that evolution - not as participants in it.*

11 References

The following sources were reviewed in the preparation of this paper. All citations are to publicly available materials as of June 2026.

[1] Wang, C.L., Singhal, T., Kelkar, A., and Tuo, J. (2025). MI9: Agent Intelligence Protocol: Runtime Governance for Agentic AI Systems. Barclays / Columbia University. arXiv:2508.03858 [cs.AI]. https://arxiv.org/abs/2508.03858

[2] Siddique, I. et al. (2026). Governance at the Speed of Agents: Microsoft Agent Framework and Agent Governance Toolkit, Better Together. Microsoft Developer Blog. Published May 14, 2026. https://devblogs.microsoft.com/agent-framework/governance-at-the-speed-of-agents-microsoft-agent-framework-and-agent-governance-toolkit-better-together/

[3] Pusukuri, K. (2026). Runtime Governance for Enterprise Agentic AI. Oracle Cloud Infrastructure Blog. Published April 23, 2026. https://blogs.oracle.com/ai-and-datascience/runtime-governance-enterprise-agentic-ai

[4] U.S. Food and Drug Administration. (2022, amended 2025). Requirements for Additional Traceability Records for Certain Foods (Food Traceability Final Rule). FSMA Section 204(d). Compliance date extended to July 20, 2028. https://www.fda.gov/food/food-safety-modernization-act-fsma/fsma-final-rule-requirements-additional-traceability-records-certain-foods

[5] European Parliament and Council of the European Union. (2024). Regulation (EU) 2024/1689 -- Artificial Intelligence Act. Article 12: Logging Capabilities and Tamper-Evident Audit Trail Requirements. https://artificialintelligenceact.eu/article/12/

[6] Kaptein, M., Khan, V.-J., and Podstavnychy, A. (2026). Runtime Governance for AI Agents: Policies on Paths. Eindhoven University of Technology / Kyvvu B.V. arXiv:2603.16586 [cs.AI]. https://arxiv.org/abs/2603.16586

[7] Oracle Corporation. (2026). Oracle Introduces Fusion Agentic Applications. Press release, March 24, 2026. https://www.oracle.com/news/announcement/oracle-introduces-fusion-agentic-applications-2026-03-24/

[8] KPMG LLP. (2026). Enterprise AI Agent Deployment Survey. Cited in Kaptein et al. (2026), arXiv:2603.16586.

[9] European Parliament and Council of the European Union. (2024). Regulation (EU) 2024/1689, EU Artificial Intelligence Act. Full text. https://artificialintelligenceact.eu/

[10] ServiceNow. (2026). ServiceNow expands AI Control Tower to discover, observe, govern, secure, and measure AI deployed across any system in the enterprise. Knowledge 2026 press release. May 2026. https://newsroom.servicenow.com/press-releases/details/2026/ServiceNow-expands-AI-Control-Tower-to-discover-observe-govern-secure-and-measure-AI-deployed-across-any-system-in-the-enterprise/default.aspx

12 Conclusion and Invitation

The enterprise AI governance gap described in this paper is real, documented, and growing. It is not fully addressed by any reviewed software category. It is not a product-marketing claim. It is a structural consequence of multi-system enterprise architecture operating at the speed and autonomy of AI-driven workflows.

The architectural framework introduced here - continuous runtime governance across the full transaction path - represents one response to that gap. It is not the only possible response. But the principles it embodies - cross-system context aggregation, action-based constraint evaluation, inter-step collision detection, governed interruption, human-in-the-loop resolution, and cross-system audit persistence - are the minimum requirements for any governance architecture that takes the problem seriously.

HimalAIan, LLC™ is actively developing this framework and seeking design partners for initial implementation. Initial focus areas include perishable goods distribution, eCommerce fulfillment, and enterprise integration-intensive environments. If your organization operates autonomous AI agents across multi-system enterprise workflows and recognizes the governance gap described here, we would welcome a conversation.

Michael Mallon
Founder, HimalAIan, LLC™
michael.mallon@himalaian.com

Patent Pending: HIMA101PR (App. No. 64/067,005) and HIMA102PR (App. No. 64/076,093)
HimalAIan, LLC™ · Meridian, Idaho · michael.mallon@himalaian.com · himalaian.com