Human Interaction and Emerging Technologies (IHIET 2026), Vol. 236, 2026, 610–621
https://doi.org/10.54941/ahfe1008136

Governing Agentic AI in Enterprise Operations: Architectural 'Rails' for Safe, Deterministic, and Compliant Autonomous Systems

Elizabeth Koumpan[1] and Vimal Dimpi[2]

[1]IBM, BTO Business Consulting Services, ON, Canada
[2]IBM, BTO Business Consulting Services, NY, USA

ABSTRACT

As enterprises accelerate the adoption of autonomous and agentic AI, the need for robust governance has become a critical architectural priority. Large organizations operate under strict regulatory, operational, and financial constraints, where even a single incorrect payment, billing error, or missed reconciliation can lead to significant compliance violations, audit failures, and material financial losses. These environments depend on deterministic, traceable, and verifiable execution; therefore, AI-driven automation cannot operate freely but must be deployed on well-defined "rails" that enforce consistency, accountability, and operational safety. This paper argues that the introduction of agentic AI requires a substantial expansion of traditional enterprise architecture principles to address new behavioral, security, and governance risks emerging from non-deterministic AI systems interacting with heterogeneous operational platforms-ERP, HCM, CLM, asset management, workflow engines, and domain-specific applications. We propose a governance-centered framework for safe agentic AI in enterprise settings, emphasizing lifecycle oversight (model management, testing, deployment, rollback), cross-system policy enforcement, and auditable decision lineage. Central to this framework is a security model grounded in Just-In-Time (JIT) and Just-Enough-Access (JEA) permissions, ensuring that AI agents receive only the minimal privileges required, only when needed, and never with long-standing or system-wide access. Additional safeguards include least-privilege design, segmentation boundaries, continuous audit trails, agent identity isolation, controlled inter-agent communication, and human-in-the-loop escalation for high-risk or sensitive tasks. These controls prevent unauthorized lateral movement, protect sensitive financial and HR data, and ensure agent actions remain aligned with organizational risk and compliance boundaries. By integrating these governance mechanisms with orchestration and policy engines, enterprises can achieve predictable execution, transparent reasoning, and resilient automation at scale. This work highlights why governance is not peripheral but foundational to the safe deployment of agentic AI.

Keywords: AI Agents, Governance, Agentic framework, Human-in-the-loop

Received July 27, 2026; Revised August 5, 2026; Accepted August 13, 2026; Available online August 25, 2026

INTRODUCTION

AI fundamentally altered the automation landscape for enterprises. Where earlier generations of automation, such as robotic process automation (RPA), business process management (BPM), and decision engines, followed explicitly programmed, deterministic instruction sequences, modern AI agents reason across ambiguous inputs, dynamically select tools, generate intermediate plans, and execute multi-step workflows without line-by-line human direction. This shift from rule-following to goal-pursuing automation introduces transformative capabilities for enterprise operations, where AI agents can handle exceptions, synthesize insights across disconnected systems, and complete complex cross-functional tasks at machine speed.

However, the same characteristics that make agentic AI powerful, also now make it highly challenging to govern. Non-determinism means the same agent, given the same input on two separate occasions, may reason differently and choose different actions. Goal-directed behaviour means agents may pursue objectives through unexpected pathways if constraints are not explicitly encoded. Using tools and APIs, agents can directly interact with production systems, like ERP records, HR databases, financial ledgers, contract repositories, where unauthorized actions threaten both operational integrity and regulatory compliance. In a multi-agent collaboration network, a single vulnerable agent can become a source of cascading errors across an entire ecosystem of interconnected agents.

Traditional enterprise architecture principles, such as separation of duties, least-privilege access, audit trails, change control, were designed for human actors and deterministic software. They remain necessary but are no longer sufficient. Agentic AI systems require a new set of architectural 'rails': governance mechanisms that constrain, observe, and verify agent behaviour continuously, not just at deployment time. Without these rails, enterprises face a new category of operational risk: autonomous systems that are technically functional but ungovernable without required accountability structures.

This paper proposes eight foundational governance principles for enterprise agentic AI , focusing on detailed security model grounded in agent identity and Just-In-Time/Just-Enough-Access (JIT/JEA) permission management. We illustrate these principles through the lens of a Talent Recruiter Digital Assistant, a representative use case where agents must interact with highly sensitive HR data, navigate employment law obligations, and produce decisions that directly affect individuals and carry legal accountability.

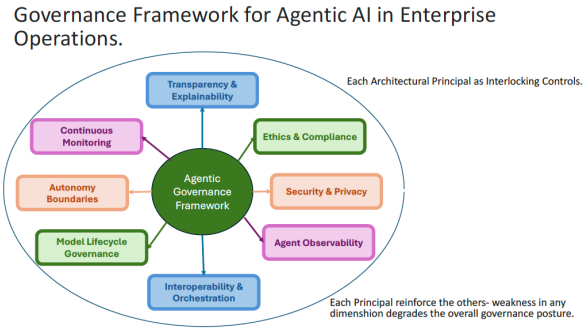

Figure 1: The eight governance principles for enterprise agentic AI (IBM owned).

Developing the Governance Framework

Managing AI systems requires a multi-dimensional governance framework. We organize this framework around eight architectural principles that together define the 'rails' within which safe agentic deployment is possible. These principles are not independent controls but an integrated system, where weakness in any one dimension degrades the effectiveness of the others.

Transparency and Explainability

While operating in enterprise workflows, AI agents must be able to provide human-readable rationale for their decisions and actions. This requirement is a precondition for regulatory compliance in domains such as financial services (where adverse action notifications are legally required), employment law (where algorithmic hiring decisions face discrimination scrutiny), and public sector operations (where audit rights attach to procurement and payment decisions).

Transparency must be architecturally enforced. Every agent action should be anchored by a structured decision trace: the data retrieved, the policies applied, the logic executed, and the discarded alternatives. These traces will support compliance evidence for auditors, debugging information for engineers, and oversight data for risk managers. Modern orchestration frameworks such as IBM Watson Orchestrate, LangGraph, and AutoGen support structured reasoning capture; governance frameworks must mandate its use and define the retention and accessibility requirements for reasoning logs.

We treat cost observability as an architectural principle rather than an operational afterthought. In other words, every agent action should record its consumption alongside its decision trace, supporting the following measures:

- cost per agent action
- cost per completed task and per resolved exception
- cost per human-in-the-loop escalation
- cost per delegation chain aggregated across sub-agents cost attributed to a named business process & in multi-tenant operations/ client

Explainability in inter-agent communication is essential. Downstream agents must fully trace every delegation chain back to its source, including which orchestrator issued the instruction, the human who authorized it & its stated purpose.

Ethics and Regulatory Compliance

Agentic systems operating in talent management, financial processing, and customer-facing applications must encode compliance requirements as first-class architectural constraints, with compliance validation occuring before action execution, not after. A payment agent must verify policy compliance before initiating a transfer; a recruiting agent must validate screening criteria

against employment law before filtering candidates; a contract agent must confirm authorization levels before committing to vendor terms.

Bias detection requires systematic intervention across data, models and decisions. First, data curation must document provenance and apply fairness audits. Next, model evaluation must include disparate impact testing for protected classes. Finaly, runtime decisions must be monitored to catch emergent bias.

When utilizing AI in recruitment, automated candidate ranking must rely on transparent, job-related criteria, with meaningful 'human-in-the-loop' checkpoints before taking an adverse employment action, ensuring full compliance with evolving legislation.

Security and Privacy

Securing agentic systems demands a fundamentally different threat model than traditional application security. Unlike conventional software that passively consumes data, AI agents are active operational actors, equipped with credentials, tool access, and authority to execute transactions across production systems. A compromised agent is a breached executor capable of initiating payments, modifying records, and escalating privileges at machine speed.

Privacy controls must be embedded at the data access layer: agents operating on personally identifiable information (PII) or regulated data must apply anonymization, pseudonymization, or tokenization before processing, and must be architecturally prevented from persisting or transmitting raw sensitive data outside approved boundaries. Agent digital identities, discussed in detail in Section 3, must be isolated, non-transferable, and scoped to the minimum permissions required for each specific task. To secure inter-agent communication, all messages must use encrypted, authenticated channels, that verify data integrity. This directly prevents prompt ingestion attacks, where malicious actors try to insert unauthorized instructions into the agent context.

Agent Observability

Observability for agentic systems requires full visibility into what each agent did, what data influenced its decisions, which tools it invoked, what it returned, and how downstream agents or systems reacted to its outputs. This end-to-end decision lineage is the foundation of accountability in autonomous systems.

Observability frameworks generally presume a single trust boundary. Enterprise agentic deployments, particularly in outsourced and managed-services operations, rarely have one. Where a provider operates business processes on behalf of a client, using the client's data, under regulatory obligations the client cannot delegate, the question of where an agent physically executes becomes a governance decision rather than an infrastructure preference. In our view, three topologies recur in practice:

Topology	Principal Trade-Off
Client-resident	Governance consistency across clients estate is difficult to achieve & the provider cannot demonstrate uniform control operation
Provider-resident	Maximizes governance consistency & unit economics; client data crosses the boundary are difficult for the clients to discharge
Federated	Preserves sovereignty & governance consistency simultaneously. This will require a verifiable isolation proof & carries higher architectural complexity

This requires structured logging with immutable timestamps to record every action, hierarchical trace identifiers to map multi-agent workflows, behavioural anomaly detection to identify deviations, and real time operational dashboards for situational awareness at scale. Observability data must be stored in tamper-resistant, append-only systems to guarantee data integrity.

Agent Interoperability and Orchestration

As enterprises deploy multiple specialized agents across different domains and platforms, interoperability becomes a governance requirement. Agents built on different frameworks, using different underlying models, and managed by different teams must communicate through standardized, governed protocols.

Emerging standards such as the Model Context Protocol (MCP) and Agent-to- Agent (A2A) establish a foundation for AI systems to securely interact. They allow AI agents to discover what tools are available, negotiate permissions and transfer contextual data using standardized, trackable format.

Governance frameworks for AI systems should strict communication standards, including limits on message size and timeout thresholds. Orchestration platforms should be configured to enforce these standards and reject non-compliant communications.

Model Lifecycle and Governance

AI agents are dynamic systems, not static software. Their underlying models drift as operational data shifts away from training distributions, while evolving APIs after tool integrations. Furthermore, shifting operational contexts can degrade their optimized performance. Model lifecycle governance must address this continuous evolution with the same rigor applied to production software: version control, change management, testing gates, deployment approvals, and rollback capabilities.

Training data must be curated with quality metrics, and bias assessments. Model versions must be immutably recorded with their training configurations, evaluation results, and approval histories. Deployment must proceed through staged environments with defined promotion criteria. Runtime monitoring must detect performance degradation and trigger automated rollback or human review when thresholds are breached. This lifecycle discipline is

the mechanism by which organizations maintain accountability for agent behaviour over time, including demonstration of the governance applied to a specific model version at a specific point in time.

Autonomy Boundaries and Human-in-the-Loop

Not all agent decisions carry the same weight. A robust governance framework must calibrate autonomy based on an action's financial materiality, legal sensitivity, and reversibility.

For the Talent domain, this might mean: autonomous scheduling of screening interviews (low risk, reversible), autonomous generation of interview feedback summaries for recruiter review (medium risk, human reviews output before action), and human-required decision on offer extension (high risk, irreversible, legal accountability). Human-in-the-loop checkpoints must be architecturally enforced, with workflow gates that require confirmed human approval before high-risk agent actions are executed.

Continuous Monitoring and Feedback

Governance does not end at deployment. We implement continuous monitoring that assesses both technical performance and regulatory governance. By uniting quantitative metrics with human-in the-loop insights, we build a closed-loop system where operational experience actively drives model refinement.

We should configure monitoring systems to detect emergent risk patterns: agents exhibiting unusual access patterns, inter-agent message volumes that deviate from baselines, decision distributions that shift unexpectedly, or escalation rates that suggest model degradation. Quarterly governance reviews should synthesize monitoring data into risk assessments that inform decisions about model updates, autonomy boundary adjustments, or agent retirement.

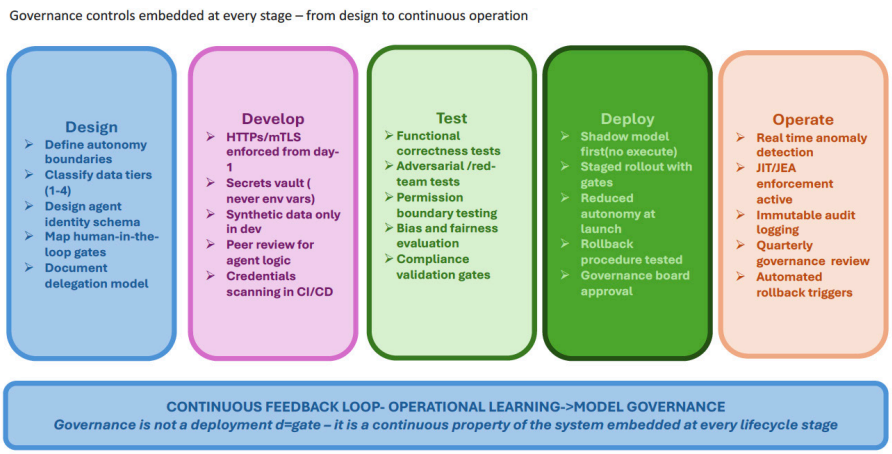

Figure 2: Secure agentic AI development lifecycle (IBM owned).

Agent Identity – The Foundation od Agentic Security

Of all the governance controls discussed in this paper, agent identity is the most foundational, but also the most commonly neglected. In current practice, many enterprise deployments allow AI agents to operate under the identity of the human user who invoked them, inheriting that user's credentials, permissions, and audit attribution. This approach is expedient but dangerous: it breaks audit trails (actions appear in logs as human actions when they were performed autonomously), enables privilege escalation (agents inherit permissions far broader than their specific task requires), and creates accountability gaps (when an agent takes an incorrect action under a human identity, responsibility is obscured).

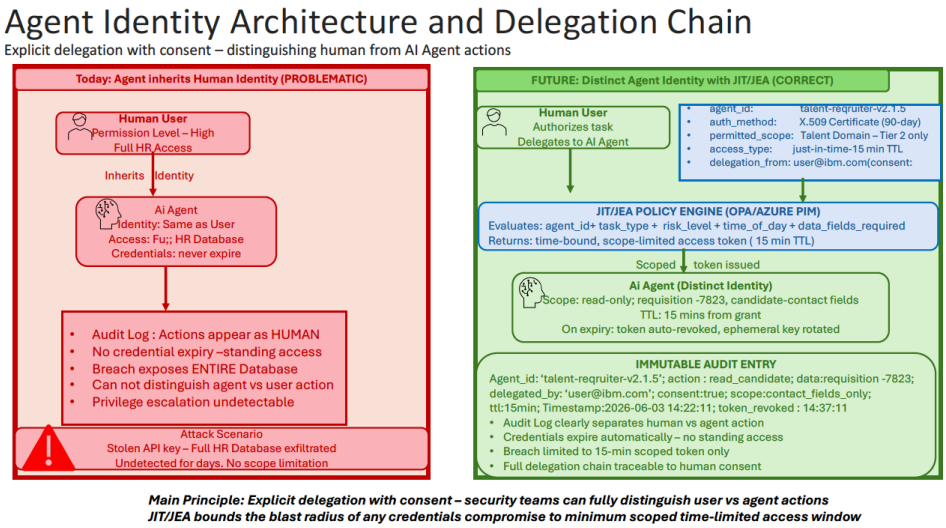

Figure 3: Comparison of the problematic pattern (agent inherits human identity) versus the correct architecture (distinct agent identity with JIT/JEA) (IBM owned).

The Case for Distinct Agent Identities

Every AI agent deployed in an enterprise environment should have a distinct, non-human identity, a cryptographically verifiable credential that uniquely identifies the agent, its version, its permitted capabilities, and its deployment context. This identity is distinct from the human identity, who invoked the agent and from the identity of the orchestration platform through which it operates. An AI agent's identity serves as the foundation of our governance framework: permissions are granted directly to the agent rather than inherited from the human who invokes it. Audit logs attribute actions to the agent identity, with human delegation chains recorded separately, while security monitoring continuously tracks behaviour to detect anomalies.

Our guidance on agentic AI governance articulates this principle directly: organizations should implement explicit delegation with consent, ensuring the user authorizes the agent to perform actions and the system records that delegation.

Agent Identity Architecture

A production-grade agent identity architecture requires several components working in concert. Each agent is provisioned with a unique identifier in a centralized agent registry, analogous to an enterprise identity provider but specifically designed for non-human principals. The registry records the agent's name, version, owning team, permitted capabilities, data access tiers, deployment environment, and lifecycle status.

Authentication is performed through certificate-based mechanisms (X.509 certificates issued by an enterprise certificate authority) or mutual TLS, providing cryptographic assurance that the agent presenting a credential is the agent it claims to be. Short-lived credentials, with lifetimes measured in hours or minutes rather than months—reduce the window of exploitation if credentials are compromised. Automated rotation ensures that credential expiry does not create operational gaps.

Authorization operates through attribute-based access control (ABAC) policies that evaluate not only the agent's identity but its operational context: what task has been delegated to it, by which human principal, under which authorization level, at what time, and for what system. This contextual evaluation enables fine-grained permission scoping that is not possible with role-based access control alone.

Just-In-Time and Just-Enough-Access in Practice

The JIT/JEA model operationalizes least-privilege access for agentic systems. Rather than granting agents standing access to the systems and data they might conceivably need, JIT/JEA provides access dynamically at the moment of need, scoped precisely to the requirements of the immediate task, and revoked automatically upon completion or timeout. In regulated domains like Talent and HR, granting AI agents standing access to sensitive data and systems poses significant security and compliance risks. An agent that maintains persistent access to employee records, compensation data, or candidate PII represents an attack surface that scales with the sensitivity of the data and the duration of the access window. JIT/JEA collapses that window to the minimum necessary.

The practical implementation proceeds as follows: the agent requests elevated privileges only when it needs to perform a specific task (e.g., fetching candidate assessment data for a specific requisition). The policy engine, implemented through tools such as Open Policy Agent (OPA) or Azure PIM, evaluates the request against real-time context: the agent's identity and version, the task type and its risk classification, the specific data fields required, the time of day and access location, and any anomaly signals from the monitoring system. The policy engine returns a time-bound, scope-limited access grant. Upon task completion or expiration, the access token is automatically invalidated and the event is logged with full context. This model directly addresses the threat illustrated by the Talent domain attack scenario: an attacker who compromises an agent operating under JIT/JEA controls gains access only to the narrowly scoped, time-limited permissions active at the moment of compromise.

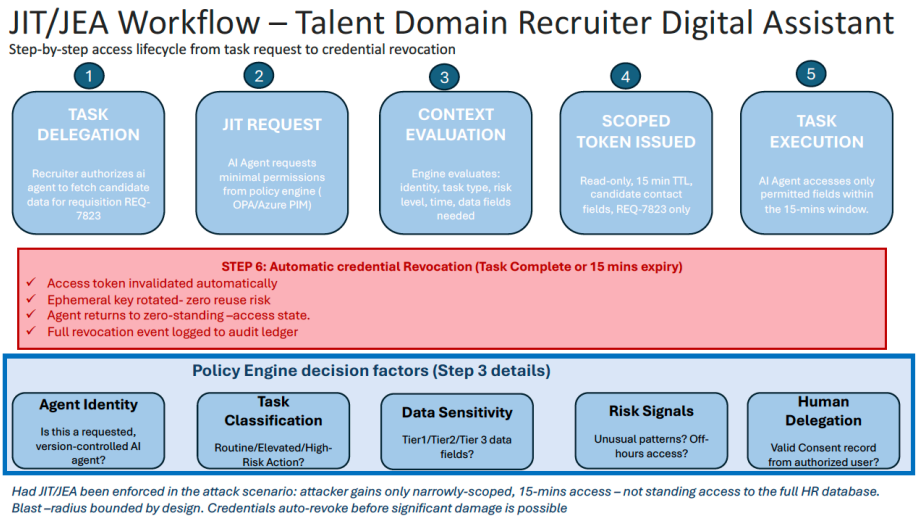

Figure 4: Step-by-step JIT/JEA access lifecycle for the talent domain recruiter digital sssistant. (IBM owned).

The Delegation Chain: Human Authorization in Multi-Agent Systems

In multi-agent architectures, where orchestrators spawn specialized sub-agents to complete delegated tasks, the question of human authorization becomes complex. Who authorized the sub-agent's actions? The orchestrator? The original human who initiated the workflow? Both?

Governance requires that human authorization propagate explicitly and verifiably through the delegation chain. When a human authorizes an orchestrator to initiate a talent screening workflow, that authorization must be recorded with the orchestrator's identity, the scope of the delegation (which actions are authorized, for which requisitions, within which policy bounds), and the identity of the human principal. When the orchestrator delegates sub-tasks to specialized agents (a CV parsing agent, a skills matching agent, a scheduling agent), each delegation must be recorded as a child authorization linked to the parent human grant.

The result is a complete, traversable delegation tree: every agent action can be traced back, through a chain of documented delegations, to a specific human authorization event. This structure satisfies both the audit requirements of regulated industries and the explainability requirements of stakeholders who need to understand why an agent took a specific action.

SECURE DEVELOPMENT, TESTING, AND DEPLOYMENT

Secure Development Environment

The security posture of an agentic AI system is established long before production deployment. Development environments that handle authentication tokens, API keys, and privileged service credentials must enforce the same security standards as production, a principle widely

violated in practice. The use of unencrypted HTTP in development environments, for example, introduces a significant security risk by exposing credentials in plain text. Even when synthetic or masked data is used in development, intercepted credentials can enable lateral movement into production environments where privileged agentic capabilities may be misused to execute transactions, modify records, trigger workflows, or extract sensitive data.

Development governance requirements include: enforced HTTPS and mutual TLS for all agent communications from the earliest development stage; secrets management through dedicated vaults (HashiCorp Vault, AWS Secrets Manager) rather than environment variables or configuration files; synthetic data generation pipelines that produce realistic but non-sensitive test data without any exposure of production PII; and separation of development, staging, and production identity providers to prevent credential reuse across environments. Code repositories must enforce branch protection, require peer review for changes to agent logic or permission configurations, and integrate security scanning tools that detect credential exposure, injection vulnerabilities, and policy misconfigurations before merge.

Testing and Validation

Testing agentic AI systems requires evaluation frameworks that go beyond functional correctness to assess governance compliance, security posture, and behavioural boundaries. A comprehensive testing for enterprise agents includes: functional testing (does the agent complete its intended task correctly?), adversarial testing (does the agent behave safely when given malicious, ambiguous, or unexpected inputs?), permission boundary testing (does the agent attempt to access data or systems outside its authorized scope?), and compliance validation (does the agent's behaviour conform to relevant regulatory requirements for the domain?).

Red-teaming exercises, in which security specialists attempt to exploit agent vulnerabilities through prompt injection, credential theft simulation, and privilege escalation attempts, should be a mandatory component of the pre-production test plan for any agent with access to sensitive data or the ability to execute consequential transactions. Bias testing, specifically examining decision distributions across protected characteristics in the Talent domain, must be documented with quantitative results before deployment approval is granted.

Deployment and Rollback

Production deployment of agentic systems must follow a staged model: initial deployment to a shadow environment where agent actions are logged but not executed (shadow mode), followed by limited production exposure with heightened monitoring and reduced autonomy levels, followed by full production deployment with standard governance controls active. Each stage transition requires documented review and approval by the designated governance authority for the agent.

Rollback capabilities must be designed and tested before deployment, not after an incident. Every agent deployment must have a documented rollback procedure that can restore the previous agent version, its permission configuration, and its associated model artifacts within a defined recovery time objective. Automated rollback triggers, activated when monitoring detects anomaly rates, error rates, or compliance violation rates exceeding defined thresholds, reduce the time between incident detection and remediation in cases where human response is too slow to prevent material impact.

CONCLUSION

Effective AI agent governance cannot rely on technical execution alone, it requires a comprehensive framework that ensures safety, integrity, trust, and operational excellence at every stage of the agent lifecycle. Because agentic systems can reason, act, and make decisions across enterprise workflows with minimal human direction, governance must be proactive, embedded, and continuous rather than reactive and periodic.

The eight architectural principles presented in this paper: transparency and explainability, ethics and regulatory compliance, security and privacy, agent observability, interoperability and orchestration, model lifecycle governance, autonomy boundaries with human-in-the-loop controls, and continuous monitoring, form a coherent framework in which each control reinforces the others.

Agent identity emerges from this analysis as the foundational control on which all others depend. Without distinct, cryptographically verifiable agent identities and explicit delegation chains, audit trails are incomplete, access controls are imprecise, and accountability is obscured. The JIT/JEA permission model operationalizes least-privilege access for agentic systems, bounding the blast radius of any compromise and eliminating the standing-access vulnerabilities that characterize naive agentic deployments.

The Talent domain illustrates with particular clarity why these controls are operational necessities. Agents that screen candidates, manage interview workflows, and support offer decisions operate at the intersection of enterprise efficiency and individual rights. Governance failures in this domain are not merely technical incidents- they are legal liabilities, reputational risks, and harms to real people. The same logic applies, with domain-specific variations, to financial processing, customer data management, healthcare operations, and every other sensitive enterprise domain in which agentic AI is being deployed.

Ultimately, governance is the foundation for AI safe scaling. Organizations that embed these principles into the architecture, development lifecycle, and operational management of their AI agents will be positioned to deploy autonomous workflows confidently, maintain trust with employees, clients, and regulators, and accelerate innovation without compromising control, compliance, or ethical standards. The measure of successful agentic AI is not what the agent can do, but what the organization can trust it to do, repeatedly and safely, at enterprise scale.

ACKNOWLEDGMENT

The authors would like to acknowledge the contributions of the IBM BPOD and Offering Consulting Services teams, whose operational experience with enterprise AI deployments informed the governance framework presented in this paper.

REFERENCES

Anthropic. (2024). Claude's Model Specification: Responsible Deployment of AI Agents. Anthropic PBC.

Canadian Centre for Cyber Security. (2025). Cloud Security Control Profile: Protected B / Medium Integrity / Medium Availability. Government of Canada.

European Parliament. (2024). EU Artificial Intelligence Act. Official Journal of the European Union.

IBM Institute for Business Value. (2024). AI Governance in the Enterprise: From Principles to Practice. IBM Corporation.

IBM Research. (2024). Agentic AI Security: Identity, Delegation, and Least-Privilege Access in Multi-Agent Systems. IBM Technical Report.

LangChain, Inc. (2024). LangGraph: Multi-Agent Orchestration Framework. https://langchain-ai.github.io/langgraph

Microsoft. (2024). Responsible AI Standard, v2: General Requirements. Microsoft Corporation.

National Institute of Standards and Technology. (2023). NIST AI Risk Management Framework (AI RMF 1.0). U.S. Department of Commerce.

Open Policy Agent. (2024). Policy-as-Code for Cloud Native Environments. https://www.openpolicyagent.org

SAP SE. (2025). SAP Sovereign Cloud for Canada: CCCS Cloud Assessment Summary. SAP Canada News Center.

Sookman, B. (2025). Canadian Data Sovereignty: Legal Considerations for Enterprise Cloud Deployments. McCarthy Tétrault LLP.

Treasury Board of Canada Secretariat. (2023). Directive on Automated Decision-Making. Government of Canada.