Designing Accountability In: Technical Architecture and Human Oversight Across AAMM Levels

Paper 3 in a Series on AI Governance and Organizational Authority

---

Alexander Huseby
Founder, Cognitive Liberty Institute
Oslo, Norway — alexander@coglib.no
2026

Abstract

The first two papers in this series established that AI decision-making authority is expanding across organizations faster than governance frameworks can keep pace, and applied the AI Authority Maturity Model (AAMM) to documented enterprise deployments to demonstrate the consistency of this pattern. This paper addresses the underlying technical question: how does system architecture itself enable or undermine human oversight, and what specific engineering practices are required to maintain accountability at each AAMM level? The paper introduces a distinction between governance (organizational policy) and guardrails (technical enforcement), and argues that this distinction is consequential: governance without guardrails is unenforceable, while guardrails without governance are arbitrary. Drawing on the NIST AI Risk Management Framework (2023), the EU AI Act's Article 14 human oversight requirements, the 'Moffatt v. Air Canada' (2024) liability ruling, and documented software engineering patterns in agentic AI systems, the paper maps specific technical mechanisms — domain bounding, human-in-the-loop interrupt architecture, circuit breakers, immutable audit trails, and output guardrails — to the AAMM levels at which they become necessary rather than optional. The central argument is that accountability must be designed into AI systems at the architectural level, not retrofitted after deployment, and that the choice of architecture is itself a governance decision with measurable legal and operational consequences.

Keywords: AI system architecture, human-in-the-loop, HITL, guardrails, audit trail, circuit breaker, domain bounding, AAMM, agentic AI, software engineering, AI accountability, NIST AI RMF, EU AI Act

---

1. Introduction

The first two papers in this series established the AAMM as a framework for classifying AI decision-making authority, demonstrated through case analysis

that enterprise AI deployments are advancing from Level 1 toward Levels 2 and 3 faster than accountability architectures are keeping pace, and presented empirical evidence that board-level AI governance expertise generates a measurable financial premium. Those papers addressed governance at the organizational and strategic level. This paper descends to the engineering level.

The governance-deployment gap identified in the first two papers has a technical dimension that the preceding analysis did not fully address. Governance policies — board oversight requirements, accountability frameworks, audit mandates — are only as effective as the technical systems designed to implement them. An organization that mandates human oversight of AI decisions but deploys AI systems with no technical mechanism for interrupting autonomous execution has governance in name only. Conversely, an organization that embeds oversight mechanisms directly into system architecture makes those mechanisms structurally difficult to bypass, regardless of whether a policy exists requiring them.

This paper makes three contributions. First, it introduces a precise distinction between governance (organizational policy and accountability structures) and guardrails (technical mechanisms embedded in system architecture), and argues that neither is sufficient without the other. Second, it maps specific technical mechanisms to AAMM levels, identifying what becomes architecturally necessary as AI authority expands from Level 0 through Level 3. Third, it examines a documented case — Moffatt v. Air Canada (2024) — as an illustration of what happens when accountability is absent from both governance and architecture simultaneously, and draws engineering lessons from that failure.

The paper is written from a software engineering perspective, engaging with the technical literature on agentic AI architecture, human-in-the-loop systems, and AI safety engineering. It draws on the NIST AI Risk Management Framework (NIST, 2023) and the EU AI Act's Article 14 human oversight requirements as regulatory reference points, not as comprehensive treatments of the regulatory landscape, which is addressed in the first paper of this series.

2. Governance and Guardrails: A Necessary Distinction

A critical conceptual confusion pervades both practitioner and academic discussions of AI oversight: the conflation of governance with guardrails. These are distinct concepts that operate at different levels and require different forms of expertise to implement.

Governance refers to the organizational policy framework: the decisions about what is acceptable, who is accountable, how AI systems are monitored and audited, and what happens when something goes wrong. Governance is the strategy. It is implemented through board decisions, risk committees, policy documents, training requirements, and organizational accountability structures. Governance can exist without any technical implementation — it is, at its core, a human institutional structure.

Guardrails refer to the technical mechanisms embedded directly into AI system architecture, APIs, and interfaces to enforce policy decisions in real time.

Guardrails are the enforcement layer. They are implemented by engineers in code: input filters that constrain what queries a system will process, output validators that check responses before delivery, domain bounds that prevent a system from acting outside defined parameters, interrupt checkpoints that pause execution pending human approval, and circuit breakers that halt execution when defined thresholds are exceeded.

The distinction matters because the failure modes are different. Governance without guardrails produces policies that cannot be enforced: a board that mandates human oversight of AI-initiated transactions but deploys systems that execute transactions without any technical pause mechanism has governance on paper and no oversight in practice. Guardrails without governance produce arbitrary technical constraints: a system that blocks certain output categories based on pattern matching, without any organizational decision about which categories should be blocked and why, has enforcement without accountability.

The Air Canada case (Moffatt v. Air Canada, 2024 BCCRT 149), examined in detail in Section 4, illustrates the failure of both simultaneously. The company deployed a customer-facing chatbot with neither meaningful governance (no process for ensuring the chatbot's outputs were accurate) nor effective guardrails (no technical mechanism for constraining the chatbot's claims to verified policy information). The British Columbia Civil Resolution Tribunal found the company liable for negligent misrepresentation and explicitly rejected Air Canada's attempt to treat the chatbot as a separate entity responsible for its own actions. The legal outcome was determined by an absence of design — not by a specific design failure.

The NIST AI Risk Management Framework (NIST, 2023) reflects a similar architecture with its four functions: Govern, Map, Measure, and Manage. The Govern function addresses organizational policies, roles, and accountability. Map, Measure, and Manage address technical risk identification, evaluation, and mitigation. The framework explicitly treats these as complementary and iterative, not as alternatives. Organizations that implement only the Govern function without the technical functions have an AI ethics statement; organizations that implement the technical functions without the Govern function have ungoverned controls. Neither position is adequate.

3. AAMM-Level Technical Requirements

The central contribution of this paper is a mapping of technical accountability mechanisms to AAMM levels. As AI authority expands from Level 0 through Level 3, the technical requirements for maintaining meaningful human oversight change qualitatively, not just quantitatively. Mechanisms that are sufficient at Level 1 are inadequate at Level 2; mechanisms that are adequate at Level 2 become insufficient at Level 3. This section specifies what is architecturally necessary at each level, and why.

3.1 Level 0 — Analytics Tool: Transparency and Provenance

At Level 0, AI systems produce informational outputs — dashboards, reports, predictions — that humans then act upon. The AI system itself holds no decision

authority. The technical accountability requirement at this level is therefore not about constraining AI action but about ensuring that humans acting on AI outputs can do so with appropriate understanding.

The key mechanisms are data provenance (making clear what data the AI system used to generate its output), model transparency (documenting the model's known limitations, accuracy characteristics, and failure modes), and output confidence indicators (quantifying the system's certainty in its outputs so that humans can calibrate their reliance accordingly). These are not guardrails in the strict sense — they do not constrain AI behavior — but they are technical prerequisites for informed human decision-making. Their absence does not prevent the AI system from functioning; it prevents human oversight from being meaningful.

The EU AI Act's Article 13 requirement — that high-risk AI systems be designed to enable transparency so deployers can interpret their outputs — codifies this requirement in regulatory terms. At Level 0, compliance with Article 13 is the primary technical accountability obligation.

3.2 Level 1 — AI Advisor: Output Guardrails and Usage Logging

At Level 1, AI systems generate recommendations that humans are expected to evaluate and act upon. The human decision-making authority is nominally preserved, but the practical risk — well-documented in human factors research — is automation bias: the tendency for humans to accept AI recommendations without adequate scrutiny, particularly when those recommendations are presented with high confidence or when the AI system has previously performed well (Lee & See, 2004).

The technical accountability mechanisms required at Level 1 therefore focus on preventing meaningful human oversight from collapsing into nominal human oversight. Output guardrails — filters that validate AI-generated recommendations against defined constraints before presenting them to human decision-makers — are the primary mechanism. Input guardrails that constrain the scope of queries the system will process are an important complement, preventing the system from operating outside its validated domain.

The Northwell Health radiology case (Wharton, 2025) illustrates the Level 1 accountability challenge precisely. An AI diagnostic tool that achieved 93% accuracy in clinical trials showed dramatically variable real-world performance across facilities. Investigation revealed that outcomes correlated with the quality of human oversight: facilities where radiologists received training on interpreting the AI's confidence scores and understood its limitations performed significantly better than those where practitioners either over-relied on or ignored the AI's outputs. The technical mechanism (the AI system itself) was identical across facilities; the accountability architecture — the training, the interface design, the supervision protocols — differed. The outcome differed accordingly.

Usage logging — structured records of AI system queries, outputs, and human responses — is the foundational audit mechanism at Level 1. It enables

retrospective analysis of how AI recommendations are being used, identifies patterns of over-reliance or under-utilization, and provides the evidentiary basis for accountability when AI-assisted decisions are later contested. The EU AI Act's Article 12 automatic event-logging requirement for high-risk AI systems reflects this necessity.

3.3 Level 2 — Bounded Executor: Domain Bounding, HITL Interrupts, and Audit Trails

Level 2 is the critical architectural threshold. At this level, AI systems execute consequential actions autonomously within defined domains, without per-decision human approval. The mechanisms adequate for Levels 0 and 1 are qualitatively insufficient here: output guardrails that prevent bad recommendations do not prevent autonomous actions from being taken; usage logs that record what a human decided do not record what an AI did. New technical mechanisms become necessary.

Domain bounding is the architectural foundation of Level 2 oversight. A bounded executor is defined precisely by the boundaries of its domain: the parameters within which it can act autonomously, the resources it can access, the actions it can initiate, and the conditions under which it must escalate to human review. These boundaries must be enforced at the technical level — built into the system's architecture, not merely stated in a policy document. An AI procurement system that is policy-constrained to spend within a defined budget but technically capable of exceeding it is not bounded in any meaningful sense. A system that is technically incapable of initiating a transaction above a defined threshold without triggering a human approval request is.

Domain bounding is most effectively implemented through the separation of action permissions from action capabilities. The AI system may be capable of initiating a wide range of actions; permission to initiate any specific action is granted or denied by a policy engine that sits between the AI's decision layer and the action execution layer. This architectural pattern — decision-permission-execution as distinct layers — makes domain bounds structurally enforceable rather than procedurally aspirational.

Human-in-the-loop (HITL) interrupt architecture provides the mechanism for escalating to human review when the AI system reaches the boundary of its domain or when a proposed action exceeds a defined risk threshold. In agentic AI systems built on frameworks such as LangGraph, interrupts are implemented as explicit pause points in the execution graph: the system's state is serialized and persisted, execution halts, relevant information is presented to a human reviewer, and execution resumes only after a human decision is recorded (LangChain, 2024). This pattern — serialize, pause, present, decide, resume — is structurally different from a nominal human-in-the-loop arrangement in which a human is notified after the fact or presented with outputs rather than decisions.

LangGraph's interrupt mechanism distinguishes between static interrupts (triggered before defined node types) and dynamic interrupts (triggered by runtime conditions). For governance purposes, the distinction matters: static

interrupts provide predictable oversight points that can be documented and audited; dynamic interrupts provide coverage for conditions that cannot be anticipated in advance but that trigger when the system detects anomalous or high-risk states. A complete Level 2 HITL architecture requires both. Static interrupts ensure that defined high-risk action categories always receive human review; dynamic interrupts ensure that novel situations outside the system's training distribution trigger escalation rather than autonomous action.

Immutable audit trails become mandatory at Level 2. At Level 1, usage logs record human decisions informed by AI outputs. At Level 2, AI systems are making decisions and taking actions autonomously; the audit trail must record what the AI did, not what a human decided. The distinction between a guardrail and an audit trail is consequential: a guardrail attempts to prevent a bad outcome before it occurs; an audit trail enables accountability and remediation after an outcome has occurred. Both are necessary. A guardrail without an audit trail provides no accountability when the guardrail fails; an audit trail without guardrails allows failures to accumulate before they are detected.

An effective Level 2 audit trail must be forensically reconstructable: it must record not only what action was taken but what inputs the AI system used to decide on that action, what alternative actions were considered, what confidence the system had in its decision, and what HITL interrupts were triggered and how they were resolved. This level of traceability is a software engineering requirement, not a policy one: it must be built into the system's architecture before deployment, because it cannot be reconstructed retrospectively from system logs that were not designed to capture it.

3.4 Level 3 — Strategic Recommender: Circuit Breakers and Human Veto Architecture

At Level 3, AI systems shape the strategic decision space presented to human decision-makers. This is a qualitatively different accountability challenge from Level 2. At Level 2, the risk is that the AI takes a specific bounded action that exceeds its domain or causes unintended consequences; human oversight is exercised at the level of specific actions. At Level 3, the risk is that the AI structures the options available to humans in ways that systematically bias organizational outcomes, even when humans nominally retain decision authority over each individual choice. Human oversight of individual decisions does not provide accountability for the framing of those decisions.

Circuit breakers are the primary technical mechanism for preventing Level 3 systems from accumulating unchecked influence over the decision space. A circuit breaker is a runtime control that detects when a system's behavior has exceeded defined thresholds — in terms of decision scope, resource consumption, frequency of influence, or deviation from baseline — and responds by degrading to a constrained operating mode, requiring explicit human authorization to continue, or halting execution entirely. Circuit breakers are well-established in distributed systems engineering as a fault tolerance pattern; their application to AI governance extends the same principle from technical failure to authority creep.

A Level 3 AI system that generates strategic recommendations must have circuit breakers that trigger when the system's recommendations begin to exhibit systematic patterns — consistently favoring certain options, consistently excluding certain considerations, consistently optimizing for a narrow set of metrics — that suggest the system's objective function or training data is producing biased strategic framing. Detecting these patterns requires monitoring the system's output distribution over time, not just evaluating individual outputs at the point of generation.

Human veto architecture at Level 3 goes beyond the interrupt-and-resume pattern of Level 2. At Level 2, a human approves or rejects a specific proposed action. At Level 3, the governance requirement is that humans retain the capacity to reject not just individual recommendations but the framing of the decision space itself — to say not only "I reject option A" but "the set of options I am being presented with does not adequately reflect the range of possible choices." This requires a system architecture that makes the AI's option-generation process inspectable and modifiable by human principals, not just its outputs.

The EU AI Act's Article 14 human oversight requirement, while not articulated in these technical terms, captures the same intent. For high-risk AI systems, the Article requires that human overseers be able to "properly understand the relevant capacities and limitations of the high-risk AI system," "detect and address anomalies, dysfunctions and unexpected performance," and "remain aware of the possible tendency of automatically relying or over-relying on the output." At Level 3, meeting these requirements demands inspectable decision architecture, not just output review.

AAMM Level	Primary Risk	Required Technical Mechanisms
0 — Analytics Tool	Uninformed human decisions	Data provenance, model transparency, confidence indicators
1 — Advisor	Automation bias, unconstrained scope	Output guardrails, input filters, usage logging
2 — Bounded Executor	Unauthorized action, unaccountable outcomes	Domain bounding, HITL interrupt architecture, immutable audit trails, circuit breakers (basic)
3 — Strategic Recommender	Decision-space capture, authority creep	Advanced circuit breakers, human veto architecture, distributional monitoring, inspectable option generation
4+ — Governance Participant and above	Accountability void, fiduciary incompatibility	All Level 3 mechanisms plus legal framework resolution; currently no adequate technical architecture exists

Table 1. Technical accountability mechanisms by AAMM level. Mechanisms are cumulative: each level requires the mechanisms of all preceding levels plus those specified.

4. When Architecture Fails: Moffatt v. Air Canada (2024)

The case of Moffatt v. Air Canada (2024 BCCRT 149), decided by the British Columbia Civil Resolution Tribunal on February 14, 2024, provides a documented, legally adjudicated illustration of what happens when accountability is absent from both governance and architecture simultaneously. It is an AAMM Level 1 case — a customer-facing advisory chatbot operating without decision authority — that nonetheless resulted in corporate liability because the technical and organizational conditions for meaningful human oversight were not in place.

The facts are straightforward. In November 2022, Jake Moffatt visited Air Canada's website to book bereavement travel following a family death. He interacted with Air Canada's AI chatbot, which informed him that he could apply for a bereavement discount retroactively after purchasing his ticket. This was incorrect: Air Canada's actual policy required the discount to be claimed before purchase. Moffatt purchased tickets at full price in reliance on the chatbot's advice, then sought the discount, was refused, and brought a complaint. Air Canada argued that the chatbot was a "separate legal entity" responsible for its own outputs and that the company could not be held liable for information it provided. The Tribunal rejected this argument as "remarkable" and held Air Canada liable for negligent misrepresentation (McCarthy, 2024).

From an architectural perspective, the failure has a clear diagnosis. Air Canada's chatbot operated at AAMM Level 1 without the technical mechanisms required for that level. There was no domain bounding that constrained the chatbot's outputs to verified policy content. There were no output guardrails that validated the chatbot's claims against the actual policy database before delivering them to users. There was no confidence threshold below which the system would defer to human customer service rather than generate an answer. There was no usage logging that would have flagged the pattern of incorrect bereavement fare advice before it resulted in customer harm. The chatbot had the capability to generate plausible-sounding policy information; it lacked the architectural constraints to ensure that information was accurate.

The legal outcome was determined by this architectural absence. The Tribunal found that "Air Canada did not take reasonable care to ensure its chatbot was accurate" — a standard that, in technical terms, is a requirement for output validation and domain bounding. The company's attempt to treat the chatbot as an autonomous actor separate from the organization failed precisely because the chatbot was deployed as part of the company's customer-facing infrastructure, without any governance mechanism that would have assigned accountability for its outputs.

The engineering lesson is straightforward: domain bounding and output validation are not optional features for Level 1 customer-facing AI systems; they are the technical prerequisite for the company's defense against negligence claims. A chatbot constrained to output only verified information from a policy database — and programmed to defer to a human agent when a query falls outside that database — would not have generated the incorrect bereavement fare advice. The architectural choice to deploy a more capable but unconstrained system was, as the Tribunal found, a failure of reasonable care.

The Air Canada case is at the low end of the AAMM. The accountability consequences of similar architectural failures at Level 2 and Level 3 — where AI systems are executing consequential autonomous actions, not merely generating advisory text — would be substantially more severe. The legal framework for attributing liability in those cases has not yet been fully adjudicated, but the engineering principle is the same: the organization that deploys the system is responsible for the accountability architecture of that system.

5. Engineering Practices for Maintained Oversight

This section specifies concrete software engineering practices that implement the technical accountability mechanisms identified in Section 3. The practices are organized around four principles that apply across AAMM levels, with level-specific implementations.

5.1 Principle: Architecture Before Deployment

The single most consequential engineering practice for AI accountability is the timing of accountability architecture relative to system deployment. Accountability mechanisms that are designed into a system before it is deployed are structurally enforced; those that are added after deployment are procedurally enforced and correspondingly more likely to be bypassed, degraded over time, or inconsistently applied.

The implication for engineering practice is that accountability requirements must be specified as part of the system design process, not as a post-deployment governance review. This requires that engineering teams and governance stakeholders collaborate during system design — not sequentially, with governance reviewing what engineers have built, but concurrently, with governance requirements shaping architectural decisions from the outset.

In practice, this means that for any AI system operating at AAMM Level 2 or above, the following questions must be answered before system architecture is finalized: What is the exact scope of the system's action authority? What conditions trigger human escalation? What actions are technically impossible without human authorization? What must be logged, and in what format, to enable forensic reconstruction? Who has access to modify system parameters, and what authorization is required? These questions have engineering answers that must be implemented in code, not policy answers that can be added later.

5.2 Principle: Separation of Capability from Permission

A fundamental architectural pattern for AI accountability is the separation of action capability from action permission. A well-designed AI system at Level 2 or above has two distinct layers: a capability layer (what the AI system can compute or generate) and a permission layer (what the AI system is authorized to execute). These layers must be architecturally separate, so that the capability layer cannot bypass the permission layer.

This separation is implemented through policy engines that intercept action requests from the AI decision layer and evaluate them against defined

permission rules before passing them to the execution layer. The policy engine is not part of the AI system itself — it is a separate component that the AI system cannot modify. Permissions are defined by human governance stakeholders and enforced by the policy engine; the AI system operates within those permissions, not around them.

The BlackRock Aladdin Copilot architecture examined in Paper 2 of this series implements this pattern: a filtering and access control node narrows the AI system's available action space to 20-30 relevant tools from a much larger potential set, and the system cannot give investment advice outside Aladdin's defined boundaries (Rosales & Valdez, 2025). The capability layer and the permission layer are architecturally distinct, and the permission layer is governed by organizational policy rather than by the AI system.

5.3 Principle: Checkpointing and State Persistence

Human-in-the-loop oversight of autonomous AI systems requires that system state can be preserved at defined points, presented to human reviewers in interpretable form, and resumed from the persisted state after a human decision is recorded. This is technically non-trivial for agentic systems that maintain complex internal state across multiple steps of a multi-step task.

LangGraph's checkpoint architecture implements this pattern by serializing the full execution graph state — including conversation history, tool call results, intermediate reasoning, and next-step plan — to a persistent store at defined interrupt points (LangChain, 2024). A human reviewer can inspect this serialized state, understand what the system has done and what it plans to do next, modify the state if appropriate, and approve or reject the planned next step. Execution resumes from the persisted state, incorporating any human modifications.

For production deployments, state persistence must use a durable, persistent checkpointer rather than in-memory storage, because human review may take time, systems may fail during the review period, and the persisted state may need to be accessed for audit purposes after the fact. The choice of checkpointer is an engineering decision with governance implications: a checkpoint stored in a persistent, tamper-evident system is an audit record; a checkpoint stored in volatile memory is not. Redis-backed LangGraph checkpointers, for example, achieve sub-millisecond write latency for state persistence while providing durability across system failures (Redis, 2025).

"Tamper-evident" in the context of audit logging has a specific technical meaning that should not be conflated with general data security. A tamper-evident audit log is one where any post-generation modification is mathematically detectable. This is achieved through two complementary mechanisms that must both be present: Write Once Read Many (WORM) storage at the infrastructure layer, which physically prevents modification of stored records; and cryptographic hash chaining at the application layer, which generates a cryptographic signature over each log entry that is bound to the preceding entry, creating a chain where any modification to any entry breaks the chain and is detectable on verification. WORM storage alone prevents tampering but provides no proof that tampering did not occur; cryptographic chaining

alone provides proof of integrity but can be bypassed if the storage layer is compromised. Cloud-native implementations of this combination include AWS CloudTrail with S3 Object Lock, Azure Immutable Blob Storage with policy locks, and Google Cloud Storage with Bucket Lock. SOC 2 controls CC7.2 and CC7.3 require immutable storage for at least one year with time-stamped logs and evidence of continuous monitoring (Dev Journal, 2026). These are the minimum infrastructure requirements for an audit trail that satisfies regulatory scrutiny at AAMM Level 2 and above.

A critical engineering trade-off exists between audit trail completeness and performance overhead. Hardware-based tamper-evident logging — which synchronously flushes each log entry to non-rewritable storage before the system can proceed — provides the strongest integrity guarantees but introduces substantial latency under high throughput, as the system cannot continue until each write is confirmed. Software-based cryptographic chaining with asynchronous writes to WORM storage achieves a better latency-integrity balance for most enterprise deployments: writes are asynchronous (preserving throughput), but the cryptographic chain provides mathematical proof of integrity for any records that were successfully written. The residual risk — that a system failure between an action and its asynchronous log write creates an unlogged action — must be acknowledged in the system's risk documentation and mitigated through transaction design that minimizes this window.

5.4 Principle: Performance and Governance as Joint Design Constraints

One of the most significant practical challenges in implementing the accountability architecture described in this section is the performance overhead it introduces. This trade-off must be addressed explicitly at the design phase, because organizations that discover it at deployment often respond by degrading the accountability architecture rather than redesigning the system.

State serialization and persistent checkpointing at HITL interrupt points introduce latency that is not present in unconstrained autonomous execution. The magnitude depends on the checkpointer backend, the size of the serialized state, and whether writes are synchronous or asynchronous. For systems using Redis-backed checkpointers, state persistence operations can achieve sub-millisecond latency; for systems requiring write-confirmed WORM storage, latency may be significantly higher. Audit logging at every action in a high-throughput Level 2 system can compound these costs. A supply chain AI executing hundreds of procurement decisions per minute faces a qualitatively different performance constraint from a strategic advisory system generating a handful of recommendations per day.

The engineering response to this constraint is not to eliminate accountability mechanisms but to design them proportionately. Three patterns manage the latency-accountability trade-off effectively. Risk-stratified logging writes full forensic audit records only for high-consequence actions and abbreviated operational logs for routine in-domain actions, concentrating the performance cost where accountability value is highest. Asynchronous audit writes decouple action execution from log persistence, accepting the residual risk of an unlogged action in exchange for throughput; this trade-off must be explicitly documented

and the unlogged-action window minimized through transaction design. Configurable HITL thresholds implement static interrupts only for defined high-risk action categories, with dynamic interrupts triggered by anomaly detection rather than applied universally, so that routine in-domain actions flow without human approval and the performance cost of HITL is concentrated on genuinely boundary-edge decisions.

The important governance implication is that performance trade-offs must be documented and approved by governance stakeholders, not resolved unilaterally by engineering teams. A decision to use asynchronous logging is an accountability trade-off, not merely a performance optimization; it should be recorded in the system's risk documentation and reviewed by whoever has oversight authority for that deployment.

5.5 Principle: Monitoring as a First-Class Engineering Requirement

Guardrails and HITL interrupts address accountability at the level of individual actions. They do not address accountability at the level of patterns: a system that consistently behaves at the edge of its domain, or consistently generates recommendations that exhibit systematic bias, may satisfy all per-action accountability requirements while failing the deeper accountability requirement that the system remain aligned with organizational values over time.

Monitoring — the systematic observation of AI system behavior over time, not just at individual action points — is the engineering mechanism for detecting these distributional failures. Effective monitoring for AI accountability requires more than operational metrics such as latency and throughput. It requires behavioral metrics: the distribution of action types taken by the system, the frequency of HITL interrupt triggers and their resolution patterns, the ratio of boundary-edge decisions to clearly in-domain decisions, and drift in output characteristics over time relative to a validated baseline.

Circuit breakers that halt or constrain system execution when behavioral metrics exceed defined thresholds translate monitoring outputs into accountability enforcement. A circuit breaker that triggers when the system's HITL interrupt rate exceeds a defined threshold — indicating that the system is frequently operating at the edge of its domain — is a monitoring-to-enforcement pattern that provides accountability without requiring human review of every individual action. The circuit breaker implements what a policy mandates but cannot reliably enforce through human review alone.

6. The Regulatory Landscape as an Architectural Specification

The NIST AI Risk Management Framework and the EU AI Act's human oversight requirements are not merely regulatory compliance frameworks; they can be read as architectural specifications that operationalize the requirements identified in Sections 3 and 5. This section examines how the two frameworks map to the technical requirements of the AAMM.

The NIST AI RMF (released January 26, 2023; NIST AI 100-1) organizes AI risk management into four functions: Govern, Map, Measure, and Manage. The

Govern function establishes organizational accountability structures, policies, and roles — the governance layer in the governance-guardrails distinction introduced in Section 2. The Map, Measure, and Manage functions address technical risk identification, evaluation, and mitigation — the guardrails layer. The framework's design as an iterative cycle rather than a linear process reflects the same insight that informs this paper: governance and technical controls must evolve together, not be applied sequentially.

The EU AI Act's Article 14 specifies human oversight requirements for high-risk AI systems in terms that directly map to the technical mechanisms described in Section 3. The Article requires that high-risk systems be designed to enable human overseers to: understand the system's capabilities and limitations (transparency, Level 0 requirement); detect and address anomalies and unexpected performance (monitoring and circuit breakers, Level 2-3 requirements); avoid over-reliance on system outputs (output guardrails and confidence indicators, Level 1 requirement); and interpret outputs correctly and take action to prevent or limit harm (HITL interrupt architecture and human veto, Level 2-3 requirements).

Article 12 of the EU AI Act — which mandates automatic event logging sufficient for post-market monitoring and risk identification — directly corresponds to the immutable audit trail requirement at Level 2. Article 14(4)(d) specifies that overseers must be able to "interrupt the system through a 'stop' button or a similar procedure" — the circuit breaker requirement at Levels 2-3.

These regulatory requirements are not arbitrary administrative impositions; they encode the engineering requirements that systems operating at elevated AAMM levels require for accountable operation. An organization that treats EU AI Act compliance as a legal obligation to be minimally satisfied rather than an architectural specification to be implemented will likely find itself in the position Air Canada occupied: nominally compliant with some requirements, but without the deep accountability architecture that makes compliance meaningful.

A significant gap in both frameworks is their current applicability to agentic AI. Both the NIST AI RMF and the EU AI Act were primarily designed for AI systems that operate in predictable, bounded ways. Agentic AI systems — those capable of autonomous multi-step planning and execution, operating at AAMM Levels 2 and above — exhibit emergent behaviors and failure modes that the existing frameworks do not fully anticipate. NIST has acknowledged this gap in its 2024-2025 work on agentic AI evaluation and is developing supplementary guidance. The EU AI Act's applicability to agentic systems operating across multiple high-risk categories simultaneously remains to be fully resolved through implementation and enforcement.

7. Limitations

The technical mechanisms described are not universally standardized. The specific implementations discussed — LangGraph's interrupt architecture, policy engine patterns, circuit breaker designs — reflect practices that are currently documented in the practitioner and engineering literature but are not yet

standardized across the industry. Organizations building agentic AI systems will encounter significant variation in how these mechanisms are implemented, and the mapping from AAMM level to technical requirement presented here should be treated as a framework for thinking rather than a prescriptive specification.

The analysis focuses on technical feasibility, not implementation difficulty. The paper identifies what technical mechanisms are required at each AAMM level; it does not fully address the engineering effort required to implement them, the trade-offs between oversight completeness and system performance, or the organizational challenges of maintaining these mechanisms over time as systems evolve. These are significant practical considerations that future work should address.

Regulatory mapping is jurisdictionally limited. The regulatory analysis focuses on the NIST AI RMF and the EU AI Act. Organizations operating in other jurisdictions — or subject to sector-specific regulations in financial services, healthcare, or critical infrastructure — face additional or different requirements that this paper does not address.

The Air Canada case is a single legal ruling in a small-claims context. While the case illustrates the accountability consequences of absent architecture clearly, it is not binding precedent in most jurisdictions and does not address the accountability questions that arise at Level 2 and Level 3 deployments.

8. Conclusion

This paper has argued that accountability in AI systems is designed in or designed out during the system architecture phase, and that governance policies without corresponding technical enforcement mechanisms are unenforceable in practice. The mapping of technical mechanisms to AAMM levels — from data provenance and confidence indicators at Level 0, through domain bounding and HITL interrupt architecture at Level 2, to circuit breakers and human veto architecture at Level 3 — provides a framework for engineers and governance stakeholders to collaborate on accountability requirements before systems are deployed.

The Moffatt v. Air Canada ruling establishes that this is not an abstract concern: organizations are legally responsible for the accountability architecture of their AI systems, and a court has found that "I could not be responsible for what my AI said" is not a viable defense. As AI systems advance to higher AAMM levels and the consequences of their autonomous actions become more significant, the accountability architecture required to maintain meaningful human oversight becomes correspondingly more complex and more consequential.

The central practical implication for software engineers is that accountability architecture is not a feature to be added after a system is built; it is a design constraint that shapes the system from the outset. For organizations, the implication is that governance stakeholders and engineering teams must work concurrently, not sequentially, with governance requirements shaping architectural decisions before code is written. For regulators, the implication is that technical standards for AI accountability mechanisms — not just

organizational governance requirements — are necessary to make human oversight requirements meaningful.

The fourth paper in this series will examine what happens when the technical and governance frameworks described across this series encounter the specific pressures of competitive markets: how organizations navigate the tension between the speed of AI deployment that competitive dynamics demand and the careful accountability architecture that responsible deployment requires.

References

Dev Journal. (2026, May 24). Security audit logging for AI agents: Making what your agent did provable. https://devjournal0.wordpress.com/2026/05/24/security-audit-logging-for-ai-agents-making-what-your-agent-did-provable/

EU Artificial Intelligence Act. (2024). Regulation (EU) 2024/1689 of the European Parliament and of the Council. Official Journal of the European Union. https://artificialintelligenceact.eu/

Huseby, A. (2025). Governance in the age of AI leadership: From advisory systems to organizational authority. Zenodo. https://doi.org/10.5281/zenodo.20330936

Huseby, A. (2026). The AAMM in practice: Classifying AI decision-making authority across enterprise deployments. Zenodo.

Jensen, M. C., & Meckling, W. H. (1976). Theory of the firm: Managerial behavior, agency costs and ownership structure. *Journal of Financial Economics*, 3(4), 305–360.

LangChain. (2024, December 14). Making it easier to build human-in-the-loop agents with interrupt. https://www.langchain.com/blog/making-it-easier-to-build-human-in-the-loop-agents-with-interrupt

LangChain. (2025). LangGraph: Build resilient agents. GitHub Repository. https://github.com/langchain-ai/langgraph

Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance. *Human Factors*, 46(1), 50–80.

McCarthy Tétrault LLP. (2024, February 19). Moffatt v. Air Canada: A misrepresentation by an AI chatbot. https://www.mccarthy.ca/en/insights/blogs/techlex/moffatt-v-air-canada-misrepresentation-ai-chatbot

McKinsey & Company. (2025, December 4). The AI reckoning: How boards can evolve. https://www.mckinsey.com/capabilities/mckinsey-technology/our-insights/the-ai-reckoning-how-boards-can-evolve

Moffatt v. Air Canada, 2024 BCCRT 149 (British Columbia Civil Resolution Tribunal, February 14, 2024). https://canlii.ca/t/k2spq

NIST. (2023, January 26). *Artificial Intelligence Risk Management Framework* (AI RMF 1.0). NIST AI 100-1. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1

Oracle. (2026). Runtime budget guardrails for agentic AI. Oracle AI and Data Science Blog. https://blogs.oracle.com/ai-and-datascience/runtime-budget-guardrails-agentic-ai

Ostrom, E. (1990). *Governing the Commons: The Evolution of Institutions for Collective Action*. Cambridge University Press.

Redis. (2026). AI human in the loop: Production oversight patterns. https://redis.io/blog/ai-human-in-the-loop/

Rosales, B., & Valdez, P. V. (2025). *How BlackRock orchestrates $11T in assets with production AI agents*. Public engineering presentation by BlackRock AI Engineering Leads. ZenML LLMOps Database. https://www.zenml.io/llmops-database/agentic-ai-architecture-for-investment-management-platform

Russell, S. J. (2019). *Human Compatible: Artificial Intelligence and the Problem of Control.* Viking.

Singh, J. et al. (2025). Architecting human-AI systems for effective collaboration and oversight. *Systems Engineering*. https://doi.org/10.1002/sys.70024

Suchman, M. C. (1995). Managing legitimacy: Strategic and institutional approaches. *Academy of Management Review*, 20(3), 571–610.

Weill, P., Woerner, S. L., & Banner, J. (2025, December 8). AI-savvy boards drive superior performance. *MIT Sloan Management Review.* Based on MIT Center for Information Systems Research Briefing released March 20, 2025. https://sloanreview.mit.edu/article/ai-savvy-boards-drive-superior-performance/

Wharton, University of Pennsylvania. (2025, May 27). Who's accountable when AI fails? *Knowledge at Wharton.* https://knowledge.wharton.upenn.edu/article/whos-accountable-when-ai-fails/
