The Authorization Threshold

Defining the Standard for Governance in AI Systems

Edward Meyman
FERZ, Inc.
2026

Abstract

The term "AI governance" is applied indiscriminately to monitoring, alignment, content filtering, audit logging, and policy enforcement. This terminological collapse obscures a structural question: what conditions must be satisfied for an AI-initiated action to be considered governed? This paper proposes that governance is not a property of systems but of decisions, and introduces the authorization threshold as the dividing line between governed and ungoverned actions. An action is governed only if its authorization can be established prior to execution under explicit, evaluable constraints. The paper defines the canonical action frame (the minimal structure of a governable action), distinguishes evidence from proof as categories of governance output, identifies the constraint representation layer as the missing transformation in current architectures, and specifies seven necessary conditions and five completeness dimensions for governance validity. The framework is implementation-agnostic and vendor-neutral. It provides a testable standard against which any system claiming to provide AI governance can be evaluated.

Keywords: AI governance, authorization, deterministic governance, proof-carrying decisions, constraint formalization, regulatory compliance, enterprise AI, governance standard

I. Shift the Frame

The question that dominates AI governance discourse is procedural: *how do we govern AI systems?* The answers follow accordingly. Monitor outputs. Align models. Filter content. Log activity. Build dashboards. Each of these functions is useful. None of them answers the question that governance must answer.

The prior question is definitional: *what must be true for an action to be considered governed?*

This is not a difference of emphasis. It is a difference of kind. The procedural question accepts governance as a collection of capabilities. The definitional question demands governance as a set of conditions. Capabilities can be partial, approximate, and aspirational. Conditions are either met or not met. The distinction matters because AI systems now initiate actions autonomously,

operate across regulatory domains, and produce irreversible outcomes. In this environment, governance must attach to actions, not systems, and must be established before execution, not reconstructed after observation.

The current evaluation framework asks whether a system *has* governance capabilities: monitoring, alignment techniques, control mechanisms, audit trails. A better framework asks whether a specific action *satisfies* governance conditions: was it authorized, under what constraints, by whose authority, and can this be proven?

Governance is not a property of systems. It is a property of decisions.

II. The Unit of Governance

The Subject of Governance

Before governance conditions can be defined, the subject of governance must be identified. What, precisely, is governed? The instinctive answers are inadequate.

Models are too abstract. A model is a capability, not a decision. Governing a model means constraining what it can do in general. But governance liability attaches to what it did do, specifically, in a particular instance.

Users are too coarse. A user identity establishes who initiated an action. It does not establish whether this action, in this context, under these constraints, was permitted.

Policies are too static. A policy defines what should be true. It does not verify that it was true at the moment of execution.

The correct unit of governance is: **a proposed action, in a specific context, under a specific authority claim.** This is the smallest entity to which governance conditions can meaningfully attach. Anything coarser introduces ambiguity. Anything more granular fragments the decision beyond auditability.

The Canonical Action Frame

Having identified the unit, its structure must be decomposed. Every governable action resolves six elements. If any element is missing or implicit, governance becomes interpretive. Interpretation cannot produce proof.

Element	Definition
Actor	Identity of the entity initiating the action.

Authority	Under what right, delegation, or chain of command the action is performed.
Object	What is being acted upon: data, system, resource, or decision target.
Scope	The boundaries of the action: what it includes and what it does not.
Context	Environmental, temporal, and system state at the moment of evaluation.
Constraints	The applicable rules and conditions that govern this action in this context.

The Action Frame is not a schema. It is a completeness test. A governance system that cannot resolve all six elements for a given action cannot produce a complete authorization decision for that action. It may still produce a useful decision. But a useful decision is not the same as a governed one.

III. The Authorization Threshold

With the unit of governance defined and decomposed, the central question becomes: what separates a governed action from an ungoverned one? The answer is a threshold, not a spectrum.

An action is governed only if its authorization can be established prior to execution under explicit, evaluable constraints.

This definition draws a hard line. If authorization is inferred after execution, governance has not occurred. If authorization is reconstructed from logs, governance has not occurred. If authorization depends on trusting the system to have checked, governance has not occurred. Each of these may be *evidence* that something governance-adjacent took place. None of them constitutes governance.

Two clarifications prevent misapplication. First, pre-execution placement is necessary but not sufficient. A control point that sits before execution but evaluates heuristically, probabilistically, or against an incomplete constraint set is pre-execution in position but not in capability. Governance requires both: the right placement and the right evaluation. Second, the word "established" is doing critical work. Authorization is not checked. It is established: derived from defined constraints, bound to a specific action representation, and available for independent verification.

Governance begins at the point where permission must be proven before action.

IV. Two Operational Classes of Systems

Systems that participate in AI decision-making fall into two structural classes. This is not a quality ranking. It is a categorical distinction. They answer different questions, and conflating them produces confused expectations.

Class A: Interpretive and Evaluative Systems

These systems interpret context, apply rules heuristically or probabilistically, generate decisions or recommendations, and record outcomes. Their strengths are flexibility, adaptability, and usefulness for oversight and assistance. Guardrails, content filters, alignment techniques, observability platforms, and agentic security controls all belong to this class.

Their structural limitation is that they cannot establish formal permissibility. They can estimate whether an output is acceptable. They cannot prove whether an action was permitted. The difference is the difference between a security guard who uses judgment and a locked gate that requires a key. Both are valuable. Only one produces a verifiable access record. The distinction is not quality. It is category: judgment-based systems produce assessments; derivation-based systems produce proofs. No amount of improvement to judgment closes the gap to derivation.

Class B: Authorization Systems

These systems operate on formalized constraints. They evaluate a defined action representation against those constraints and produce deterministic outcomes: allow, deny, escalate, or abstain. They emit verifiable artifacts. Their key property is that decisions are *derived*, not *interpreted*. The artifact exists because the evaluation occurred. The artifact cannot exist without the evaluation.

Class B systems are less flexible than Class A. They cannot handle ambiguity gracefully. They require explicit constraint formalization, which is expensive. These are not weaknesses to be overcome. They are the cost of proof. A system that handles ambiguity by interpreting it has produced a judgment. A system that handles ambiguity by refusing to proceed until the constraint is clarified has maintained the authorization boundary.

These are not competing approaches. They answer different questions.

V. Evidence Versus Proof

The distinction between Class A and Class B systems rests on a deeper distinction that most governance discussions leave unexamined: the difference between evidence and proof.

Evidence

Evidence describes what happened. It depends on logs, traces, internal system state, and operator trust. Its use cases are audit, investigation, and debugging. Evidence is retrospective. It tells a story about a decision after the decision has been made. The story may be accurate, detailed, and useful. But it remains a narrative. The same logs can support different interpretations. The same traces can be read in different ways. Evidence explains a decision.

Proof

Proof demonstrates that a decision followed from defined constraints. It is independent of system operator, independent of mutable system state, reproducible, and verifiable by a third party. Proof is not retrospective. It is constitutive. It does not describe the decision. It defines the space within which the decision could occur. The proof artifact is produced at the moment of decision as a byproduct of the evaluation itself.

The critical implication: evidence can justify after the fact. Proof restricts the space of valid actions before execution. An evidence-producing system says: "Here is what we did and why." A proof-producing system says: "Here is what we *could* have done, given the constraints, and here is the derivation showing this action was within that space."

Most AI governance systems today produce evidence. They produce it well. They log diligently, trace comprehensively, and report clearly. But evidence without proof creates a specific vulnerability: the governance artifact can exist without the governance evaluation. A log entry can record "policy checked" without any mechanism ensuring that the check actually bound the decision. A dashboard can display compliance status derived from metadata rather than from constraint evaluation. When the artifact can be generated independently of the evaluation, the artifact proves nothing.

One clarification prevents overreach. Proof, as defined here, guarantees constraint-consistency: the decision was derived from the constraints that were in effect. It does not guarantee that the constraints themselves were correct. Correctness is a property of the policy content, which remains the responsibility of policy authors and the governance, risk, and compliance workflows that define it. Deterministic governance ensures that whatever policy was defined is the policy that was enforced. No more, no less.

Evidence explains a decision. Proof constrains what decisions were possible.

VI. The Constraint Formalization Requirement

If proof requires derivation from defined constraints, the constraints must exist in a form that supports derivation. This is the transformation layer that most governance architectures omit. Its absence is not a gap in implementation. It is a gap in category.

Policy, Representation, Evaluation

Governance operates across three layers. The first is the **policy layer**: human-readable rules expressed in natural language. Policies are inherently ambiguous, context-dependent, and subject to interpretation. This is not a deficiency. It is the nature of institutional language. A policy that says "material transactions require senior approval" is functional for human governance because humans share interpretive context. It is non-functional for computational governance because the terms "material," "transaction," "senior," and "approval" each admit multiple valid interpretations.

The third layer is the **evaluation layer**: the runtime mechanism that evaluates actions against constraints and produces verdicts. This layer is well understood. Evaluation engines exist. Solvers exist. Authorization gates exist. The technology is not the bottleneck. The bottleneck is representational, not computational.

The bottleneck is the second layer: the **constraint representation layer**. This is where human-readable policy is transformed into a canonical, unambiguous, machine-evaluable form. Without this layer, the evaluation engine has nothing formal to evaluate against. It falls back to interpretation. Interpretation may be sophisticated, context-aware, and effective. It still cannot produce proof.

The Missing Transformation

Policy formalization requires resolving ambiguities that natural language deliberately preserves. "Material" must become a threshold. "Senior" must become a role in an authority hierarchy. "Approval" must become a signed attestation with temporal validity. Each resolution is a design decision with compliance implications. Get it wrong and the constraint does not match the policy intent. Get it right but fail to version it and the constraint drifts from the policy over time.

Deterministic governance does not eliminate interpretation. It confines interpretation to the design-time formalization process, where it can be versioned, audited, and challenged. The alternative is interpretation operating silently at evaluation time, invisible to auditors and unreproducible across decisions. The constraint representation layer makes the interpretive choices explicit, frozen, and disputable. That is a different category of risk than interpretation that occurs at runtime and leaves no auditable trace.

This is not a one-time translation. Policies change. Regulations evolve. Organizational structures shift. The constraint representation layer must maintain fidelity to the policy layer while remaining evaluable by the execution layer, and it must do so across time. The semantic definitions embedded in the constraint set must be versioned, immutable at decision time, and bound to each verdict so that auditors can confirm the meaning that was operative when the decision was made.

Everyone understands policy. Everyone understands evaluation. Almost no one has a constraint representation layer. And almost no one acknowledges what its absence means.

Without constraint formalization, all evaluation is interpretive. Even when it appears deterministic.

The pipeline is clear. Systems that operate *policy → interpretation → action* are governance-adjacent. Systems that require *policy → formal constraints → deterministic evaluation → authorization* are governance-capable. The constraint representation layer is the dividing line.

VII. The Execution Boundary

A common claim in AI governance is that a system performs governance "at execution time." The claim is ambiguous and must be refined.

Governance must occur before irreversible action. This is not controversial. What requires clarification is that placing a control point before execution does not, by itself, guarantee governance. A pre-execution checkpoint that evaluates heuristically is positioned correctly but functionally insufficient. A pre-execution filter that blocks known-bad patterns is useful but does not authorize the patterns it allows.

The determining factor is not where evaluation occurs but what evaluation can prove. A system that sits before execution and can demonstrate that authorization was derived from formal constraints under a complete Action Frame has crossed the authorization threshold. A system that sits before execution and applies best-effort pattern matching has not, regardless of its position in the architecture.

Placement is where the evaluation occurs in the execution pipeline. **Capability** is what the evaluation can prove about the action. Governance requires both. Placement without capability is a checkpoint. Capability without placement is an audit.

VIII. Conditions of Governance Validity

The preceding sections establish the authorization threshold conceptually. This section operationalizes it. Rather than listing features that a governed system should have, the following defines conditions under which a decision *fails* the governance standard. The framing is deliberate. A positive list of features invites the response "we satisfy most of those." A negative test demands completeness.

A decision fails the governance standard if any of the following cannot be demonstrated:

Condition	Failure Mode
Pre-execution evaluation	The decision was evaluated after execution, or no evaluation occurred before the action took effect. Retrospective analysis is investigation, not governance.
Constraint-bound derivation	The verdict was not derived from explicit, formal rules. It was interpreted, estimated, or heuristically determined. Interpretation is judgment. Derivation is proof.
Context completeness	The evaluation did not resolve identity, authority, object, scope, context, and constraints. A verdict against an incomplete Action Frame is a verdict about something other than the action that occurred.
Non-bypassability	An alternate execution path existed that could circumvent the authorization gate. A governance boundary with a side path is not a boundary.
Determinism	Identical governed state could produce different verdicts. Governed state includes the action representation, policy version, semantic definitions, evaluator version, and all context required for evaluation. If any element of governed state is uncontrolled or unrecorded, determinism is untestable. If the outcome is not invariant under identical governed state, the system cannot provide audit-grade evidence of consistent policy application.
Verifiability	The verdict cannot be independently checked. If verification requires trusting the system that produced the verdict, it is self-attestation, not proof.
Replayability	The verdict cannot be recomputed with an identical result by a third party using the proof artifact and referenced materials, without the cooperation of the original system.

These are not features. They are conditions of validity.

IX. The Five Dimensions of Governance Completeness

Authorization establishes validity. Completeness determines whether that validity constitutes proof.

Satisfying the seven conditions above establishes that a decision was governed. But governed decisions vary in completeness. A runtime authorization gate that evaluates a subset of applicable constraints satisfies the formal conditions of governance while leaving the remaining policy domain unevaluated. The decision is governed in form but incomplete in substance. Authorization is necessary. It is not sufficient.

Five dimensions determine whether a governed decision is complete enough to constitute proof rather than partial authorization.

1. Constraint Completeness

Is the constraint set complete for the policy domain? A gate that evaluates available rules without coverage verification may authorize actions that violate unevaluated policies. Governance completeness requires auditable coverage: every applicable policy provision represented, the constraint set itself auditable for completeness. Without it, authorization is real but governance is not.

2. Semantic Precision

Are terms defined unambiguously? "High-risk" means different things in different regulatory contexts. If the meaning of a term can drift between decision time and audit time, the verdict is not reproducible. Semantic definitions must be versioned, bound to each decision, and immutable within the scope of that decision. Without semantic precision, same inputs and same rules can produce different verdicts because the definition of "material" changed between evaluation and review.

3. Policy Binding

Is the policy version cryptographically bound to the verdict? A system that evaluates "the current policy" without recording which version was current cannot answer the question: which rules governed this action? Binding requires cryptographic hashes and immutable references embedded in the proof artifact. Without it, an auditor who asks which policy governed a decision receives a pointer to today's policy. Today's policy is not yesterday's policy.

4. Temporal Binding

Is the decision anchored to a specific moment? Policies have effective dates. Evidence has validity windows. Authority chains have expiration. A verdict rendered against expired policy or stale evidence is unauthorized regardless of whether the gate returned ALLOW. Without temporal binding, a loan approved under a policy that expired two hours before application is treated as authorized. The authorization gate checked permissions. It did not check effective dates.

5. Independent Replayability

Can a third party reproduce the verdict? If the verifier is controlled by the same entity that generated the decision, the result is self-attestation, not proof. An independent party, using the proof artifact and access to referenced policies, must be able to reconstruct the verdict without cooperation from the original system. Without this dimension, the vendor says "we can replay internally." The regulator asks: "Can I replay without you?" The answer determines whether the artifact is proof or narrative.

These five dimensions are the completeness test for any authorization artifact. A decision that satisfies the seven validity conditions from the preceding section but fails any of these five dimensions is governed but incomplete. Incomplete artifacts are not proof. They are narrative with a timestamp.

X. The Authorization Artifact

If governance produces proof, proof must take a specific form. The authorization artifact is the tangible output of a governed decision. Its structure determines whether the decision can be verified, replayed, and audited.

A valid authorization artifact must bind the following elements into a single, tamper-evident object:

Binding	Requirement
Action Definition	Complete representation of the proposed action, including all six elements of the Action Frame.
Identity and Authority	Who initiated the action, under what delegation or authority chain, with cryptographic identity verification.
Constraint Set	The specific, versioned constraints evaluated, with cryptographic hash binding to the policy version in effect.
Semantic Snapshot	The versioned semantic definitions operative at decision time, ensuring term meaning is immutable for audit.
Evaluation Result	The verdict (allow, deny, escalate, abstain) and the structured evaluation trace showing which constraints passed, failed, or triggered escalation.
Temporal Anchors	Decision timestamp, policy effective window, evidence validity period, and authority chain expiration.
Integrity Mechanism	Cryptographic binding (hash, signature) ensuring the artifact has not been modified after issuance.

Without complete binding, the artifact is descriptive, not authoritative. It may record that a check occurred. It cannot prove what was checked, against what constraints, under what definitions, at what time, and by whose authority. The artifact must support two operations: independent verification (a third party can confirm the verdict) and deterministic replay (the same inputs produce the same verdict). An artifact that supports neither is a log entry. An artifact that supports verification but not replay is an attestation. Only an artifact that supports both is proof.

XI. Implications

The framework presented here does not argue for a specific technology, vendor, or implementation approach. It defines the conditions under which governance has occurred and the conditions under which it has not. The implications follow from the definitions.

Governance shifts from oversight to precondition. Under this framework, governance is not something done to AI systems after they are built. It is a condition that must be satisfied before an action takes effect. Organizations that treat governance as a review function will find themselves unable to demonstrate that individual actions were authorized at the moment of execution.

Risk shifts from probabilistic to bounded. Evidence-based governance accepts residual uncertainty: the system probably complied. Proof-based governance eliminates this category of uncertainty for governed actions: the system demonstrably complied or demonstrably did not. Risk does not disappear. But the nature of risk changes from "we cannot be sure what happened" to "we can prove what happened and dispute whether the constraints were correct."

Responsibility shifts from inferred to explicitly assigned. When authorization is derived from a formal authority chain, bound to a specific identity, and embedded in a tamper-evident artifact, the question "who approved this?" has a verifiable answer. When authorization is inferred from system behavior and log analysis, the same question produces a narrative. Narratives can be contested. Proofs can be verified.

The key question shifts from *"What did the system do?"* to *"What was it permitted to do, and how do we know?"*

XII. Raise the Standard

As AI systems gain autonomy, the governance question becomes more urgent and less forgiving. Systems that act on behalf of institutions, make decisions with legal consequences, and operate across regulatory boundaries cannot be governed by observation alone. Governance must move from interpretation to authorization. From evidence to proof. From trust to verification.

The standard proposed here is not aspirational. It is testable. For any decision claimed to be governed, five questions can be asked: Was the constraint set complete for the policy domain? Were semantic definitions versioned and bound? Was the policy version cryptographically embedded in the proof artifact? Does the artifact include temporal validity evidence? Can the decision be replayed offline, by an independent verifier, without the cooperation of the original system?

A system that satisfies all five is providing governance. A system that satisfies some is providing something less. The difference matters because the consequences of ungoverned AI actions are no longer hypothetical. They are operational, legal, and institutional.

The standard is not whether a decision can be reviewed after the fact. The standard is whether it can be proven, before execution, that the action was permitted under defined constraints. And the proof must survive independent verification: reproducible by a third party, replayable without the cooperation of the original system, binding on its own terms.

Governance begins where trust is no longer required.

References

The framework presented in this paper draws on and extends the following published research. No inline citations appear in the body text; the essay is intended to stand on its own arguments. These references provide the formal and empirical foundations for readers who wish to examine the underlying research.

[1] Meyman, E. (2025–2026). A Taxonomy of AI Governance Approaches: Distinguishing Visibility, Alignment, and Authorization. *FERZ, Inc.*

Version 1.5. Introduces the eight-category governance taxonomy, the formal definition of deterministic governance, the Five-Point Diligence Test, and the anti-laundering tests.

DOI: 10.5281/zenodo.18275969

[2] Meyman, E. (2025–2026). Versioned Meaning and Audit-Stable Ontologies for Deterministic AI Governance. *FERZ, Inc.*

Version 1.4. Establishes the formal requirements for semantic precision in governance decisions: version binding, non-retroactivity, reproducibility, and drift visibility.

DOI: 10.5281/zenodo.18328587

[3] Meyman, E. (2025–2026). Type-Theoretic Foundations of Deterministic AI Governance. *FERZ, Inc.*

Provides the formal type system underlying constraint representation and evaluation, including proof-carrying decision structures and verifier independence.

DOI: 10.5281/zenodo.18371223

[4] Meyman, E. (2025–2026). Deterministic Governance Is Multi-Dimensional. *FERZ, Inc.*

Extends the authorization framework beyond binary allow/deny to encompass constraint completeness, evidence requirements, decision economics, and safety envelopes.

DOI: 10.5281/zenodo.18902166

[5] Meyman, E. (2025–2026). Governance Laundering: How Trust-Based Imitation Undermines Deterministic AI Governance. *FERZ, Inc.*

Identifies structural patterns by which systems approximate governance claims through observability, logging, or orchestration without satisfying the authorization threshold.

DOI: 10.5281/zenodo.18746522

[6] Meyman, E. (2025–2026). Four Tests Standard (4TS): Deterministic AI Governance Conformance Specification. *FERZ, Inc.*

Version 1.0.1. Vendor-neutral conformance standard defining the Stop, Ownership, Replay, and Escalation tests for verifiable AI governance, with PCD schemas and test vectors.

Available at github.com/edmeyman/4ts-standard

FERZ, Inc. · ferz.ai · info@ferz.ai