Epistemic Capture and the Action Boundary: Corrigibility for Learned and Agentic Public Infrastructure*

A Preprint

Anivar A Aravind
Independent Researcher
Bengaluru, India
ping@anivar.net

April 2026
Revised: 11 July 2026

Abstract

This paper extends the corrigibility framework to learned and agentic systems, where deterministic rule execution is replaced by probabilistic inference. Three structural pressures constrain corrigibility in learned systems: opacity of inference (requiring LWD-R transparency: logic, weights, data, representation), concentration of training resources (compute capture), and acceleration of automated action (governance denial of service). The Action Boundary Protocol separates probabilistic inference from deterministic execution. The invariant holds: on this framework's definition, systems count as corrigible exactly when the five conditions close the loop under stochastic verification. Across the substrate transition it is the verification machinery that changes, never the conditions.

Dependency. This paper presupposes the companion paper [Aravind, 2026], which derives the five corrigibility tests (EXIT, CODE, AUDIT, GOVERN, FORK) from control theory, commons governance, and free software. The present paper does not re-derive these tests; it extends their verification methods to stochastic systems.

Keywords Epistemic Public Infrastructure · AI Governance · Corrigibility · Action Boundary Protocol · Compute Capture · LWD-R

How to Read This Paper. Corrigibility is the minimal architectural condition that keeps public systems reversible. The argument has three parts: learned systems require bounded error propagation (the conceptual invariant); five control-layer conditions extend to stochastic inference (the architectural constraints); and these constraints discriminate between AI-mediated systems (the applied evaluation). Architecture-specific risk profiles and worked open-model examples illustrate discriminative power, not exhaustive institutional evaluation.

Executive Summary

When infrastructure incorporates machine learning, it no longer merely processes transactions. It generates claims about reality. This creates **epistemic public infrastructure (EPI)**.

Traditional transparency is insufficient. Reconstruction of model behavior requires disclosure of logic, weights, data provenance, and representational schema (**LWD-R**). Practical forkability is constrained by compute asymmetry. Performance degrades under distributional shift.

The **Action Boundary Protocol** separates probabilistic inference from deterministic execution through enforceable validation layers. Without such boundaries, agentic systems exercise unbounded epistemic power.

*Companion paper (Paper I, DPI): doi.org/10.2139/ssrn.6059075

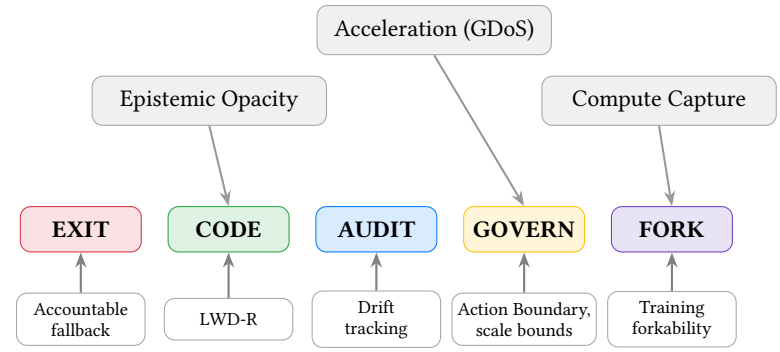

Figure 1: Framework map. Learned systems add three structural pressures (top), each degrading a specific corrigibility test (middle): epistemic opacity breaks CODE at the observability layer, acceleration overwhelm breaks GOVERN at the constraint layer (restored by the Action Boundary together with fan-out, action-rate, and blast-radius bounds), and compute capture breaks FORK at the replacement layer. The architectural responses (bottom) restore verification for each test without softening the standard. Verification methods per test are detailed in Table 3.

1 Introduction

The companion paper [Aravind, 2026] establishes a corrigibility framework for Digital Public Infrastructure (DPI), deriving five structural tests from cybernetics, commons governance, and free software. That framework addresses **deterministic** infrastructure: ledgers, registries, and payment rails where source code is the arbiter of behavior. This paper extends the framework to **stochastic** infrastructure, what we term **Epistemic Public Infrastructure (EPI)**: systems where state decisions emerge from learned parameters rather than explicit logic.

Definition 1.1 (Epistemic Public Infrastructure)**.** EPI comprises AI systems that determine citizen outcomes at population scale, including: identity verification models, welfare eligibility systems, public-facing chatbots with administrative authority, and autonomous agents operating within mandatory infrastructure.

1.1 DPI and EPI: Structural Contrast

Digital Public Infrastructure (DPI) processes transactions through deterministic logic. Epistemic Public Infrastructure (EPI) produces classifications, predictions, or inferences through learned parameters. Table 1 summarizes the structural contrast.

Table 1: Structural contrast between DPI and EPI

DPI	EPI
Deterministic rule execution	Stochastic learned inference
Logic transparency sufficient	Weights, data, and representations matter
Forkability tied to software	Forkability tied to compute capacity
Static certification possible	Performance degrades under distribution shift

Corrigibility in EPI therefore requires additional structural conditions beyond those sufficient for deterministic systems.

Tiered Adoption. EPI systems vary in consequence severity. The framework defines three deployment tiers with increasing compliance requirements (Table 2):

The tier classification is a governance decision, not a technical one. Systems must declare their tier; auditors verify appropriateness.

Gradient Quantifier in Tier Audits. Consistent with the Gradient Quantifier remark of the companion paper [Aravind, 2026], every compliance assessment under each tier is evaluated at the least-resourced sub-population within the deployment's affected class, not at the mean user. For `high_stakes` deployments, this means: exit penalty

Table 2: EPI deployment tier classification

Tier	Use Cases	Requirements
`trivial`	Translation, summarization	Action boundaries required
`decision_support`	RAG over government data	Open weights, local data storage
`high_stakes`	Welfare, policing, judicial	Full LWD-R; accountable-authority checkpoint

ratios, mean time to detect (MTTD), and override rates are reported as distributions across strata defined by literacy, language, connectivity, and administrative precarity; pass/fail determination is taken at the lower decile. A system that passes all tests for median users while failing them for the least-resourced sub-population does not pass. Auditors must document the stratum at which minimum performance was measured and the method by which that stratum was identified.

Why "Epistemic"? The term is structurally precise. In deterministic DPI, infrastructure manages *what happened*: the **Ledger State** (a payment was made, an identity was verified). In learned systems, infrastructure manages *what is true*: the **Inference State** (a person is "eligible," a transaction is "fraudulent," a citizen is a "risk"). This shift from transactional facts to administrative knowledge constitutes a transfer of **epistemic sovereignty**.

When a state uses a model to determine eligibility, it is no longer merely processing data; it is producing a *claim to knowledge*. If that claim is hidden in a latent space, the infrastructure has captured the state's ability to know or contest the truth. The "R" in the four-layer LWD-R requirement (Representation, Section 3) is the epistemic anchor: if the categories through which the state "sees" the world are proprietary and non-contestable, the state has delegated its interpretive sovereignty. Citizens are not merely locked out of a database; they are locked into a reality they cannot define.

By naming this domain **Epistemic Public Infrastructure**, the framework asserts that the "truth" produced by an AI system must satisfy the same corrigibility tests as a legal fact. The question shifts from "Does the AI work?" to "Is the knowledge produced by this system structurally falsifiable and reversible?"

Architectural Closure. DPI solves the problem of **transactions** (Scale and Velocity). EPI solves the problem of **meaning** (Stochasticity and Representation). Once the infrastructure for creating administrative reality (EPI) is bounded by the same five structural tests as the infrastructure for processing payments (DPI), the framework achieves closure. The theory has scaled from the registry to the world model.

The five structural tests are:

- **EXIT**: Reversibility of participation
- **CODE**: Inspectability of logic
- **AUDIT**: Independent verification
- **GOVERN**: Binding constraint
- **FORK**: Capacity to reproduce

The Cybernetic Foundation. These tests are not arbitrary normative preferences but derive from control theory. Ashby's Law of Requisite Variety [Ashby, 1956] establishes that a controller must possess at least as many response states as the system it governs. Applied to infrastructure: governance mechanisms must match the variety of harms a system can produce. The five tests operationalize this requirement as a closed feedback loop: EXIT provides the error signal (affected parties can refuse), AUDIT acts as the sensor (errors are detectable), CODE ensures transmission clarity (logic is inspectable), GOVERN serves as the actuator (correction is binding), and FORK provides selection pressure over the entire loop (the system can be reproduced if operators refuse correction).

The Open-Loop Instability Argument. When any test fails, the feedback loop opens. The companion paper develops this argument along two converging tracks. The control-theoretic track shows that open-loop systems drift toward divergence because errors accumulate without correction; the political-theoretic track shows that partial contestation creates *sham governance* — cosmetic accountability without corrective leverage — which is institutionally distinct from absent governance only in appearance. Neither track alone forces five-test irreducibility, but together they overdetermine it from independent grounds. A system with EXIT but no GOVERN allows users to detect harm but not correct it; a system with CODE but no FORK allows inspection but not competition. The set is irreducible because removing any one test collapses the corrective apparatus along at least one of the two tracks.

Normative Anchor. This framework adopts *non-domination* — the structural absence of arbitrary power — as the normative criterion for legitimate public infrastructure, drawing on the analytical core of Pettit's republican tradition [Pettit, 1997]. Alternative frameworks reach similar conclusions through different routes: Habermas's deliberative theory [Habermas, 1996] centers communicative legitimacy; Sen's capability approach [Sen, 1999] centers substantive freedoms; Mouffe's agonistic democracy [Mouffe, 2000] centers the productive maintenance of conflict. We center non-domination because its focus on structural protection provides the most exact isomorphism with system architecture: a digital system is public to the extent that those it governs possess the structural capacity to override its arbitrary application. The five corrigibility tests operationalize the engineering conditions under which this arbitrariness is bounded. While the framework's mechanics are compatible with overlapping democratic traditions, its foundational metric is the structural distribution of power rather than the optimization of administrative utility.

This paper stress-tests the framework against learned systems: infrastructure where behavior is determined by parameters learned from data (AI/ML) rather than explicit code logic. If the five tests are truly fundamental, they must apply not only to deterministic infrastructure but also to stochastic systems where traditional verification assumptions collapse.

The Challenge. Learned systems dismantle the assumption that source code is the sole arbiter of behavior. In AI, identical inputs may yield stochastic outputs; behavior is defined by training data; and opacity is often inherent rather than contrived. Yet the collapse of deterministic transparency does not invalidate corrigibility's structural necessity. As systems become less intelligible, external correction becomes *more* critical.

Thesis. The five corrigibility tests remain invariant for learned systems. What changes is the verification method and the resource barriers to compliance.

Scope. This paper addresses Epistemic Public Infrastructure (Definition 1.1): AI systems deployed at population scale with administrative authority. It does not address consumer AI products where exit is trivial, nor research systems without deployment impact.

Paper Structure. Section 2 establishes that the five tests remain invariant. Section 3 extends the CODE test via LWD-R disclosure and introduces the strict interpretability firewall. Section 4 introduces Variety Drift. Section 5 analyzes Compute Capture. Section 6 examines agentic scaling, GDoS, and citizen-side GOVERN mechanism families. Section 7 addresses workflow sovereignty and WCC. Section 8 provides architecture-specific risk analysis. Section 9 answers two anticipated objections. Section 10 separates corrigibility from epistemic competence. Section 11 discusses implications and limitations. Section 12 sets out deployment regimes outside the framework's current scope.

Key Contributions.

1. **Epistemic Public Infrastructure (EPI)**: Distinguishes AI-mediated administration from deterministic DPI
2. **LWD-R**: Four-layer transparency (Logic, Weights, Data, Representation)
3. **Ontological Capture**: When a model's categories become non-contestable facts
4. **Variety Drift**: Why AI corrigibility is a persistence condition
5. **Compute Capture**: When "open weights" fails FORK
6. **GDoS** (governance denial of service): Saturation of governance capacity under agentic scaling
7. **WCC**: Quantifies epistemic delegation risk
8. **Action Boundary**: Deterministic envelope around stochastic inference

2 The Invariance of the Standard

Proposition 2.1. The five corrigibility tests remain invariant for learned systems. What changes is the verification method.

The structural requirement for corrigibility derives from Ashby's Law of Requisite Variety: a controller must have at least as many response states as the system it governs [Ashby, 1956]. This requirement is independent of whether the controlled system is deterministic or stochastic. An AI model that affects citizen outcomes at population scale generates at least the variety of potential harms of deterministic infrastructure — the argument only strengthens if it generates more; therefore, the governance apparatus must possess at least equivalent corrective variety.

The proposition carries a falsifiability commitment. Every failure mode this paper catalogues is expressed as the degradation of one or more of the five conditions, and each mechanism the paper introduces is derived from that

projection rather than appended as a new principle. A corrigibility failure that could not be so expressed would refute the completeness of the set.

The table below illustrates how the mechanical verification of each test adapts to the probabilistic nature of AI without softening the structural requirement.

Test	DPI Verification	EPI Verification	Key Addition
EXIT	Can refuse the system	Disclosure + non-AI alternative; appeal to an accountable authority independent of the deciding system	Accountable fallback
CODE	Source code determines behavior	LWD-R: Logic, Weights, Data, Representation layers disclosed	**R** (ontology)
AUDIT	Inspect logic and outputs	Statistical bounds + continuous monitoring; variety drift measurement	Drift tracking
GOVERN	Maintainer and RFC process	Governance over objectives, value tradeoffs, and category definitions	Ontology governance
FORK	Copy code and deploy	Resource forkability: compute + data + training pipeline	Compute access

Table 3: Verification methods for DPI (deterministic) vs. EPI (learned) systems. The standard remains invariant; verification adapts.

Structural Pressures in Learned Systems. Three additional pressures constrain corrigibility in learned systems, and Figure 1 maps each to the test it degrades and the response that restores it. Epistemic opacity breaks CODE at the observability layer and is answered by LWD-R transparency. Compute capture breaks FORK at the replacement layer and is answered by training forkability. Acceleration overwhelm (GDoS) breaks GOVERN at the constraint layer and is answered by the Action Boundary.

Deployment Variables. The invariance thesis extends past verification methods to deployment geometry. Where the accountable authority sits (in the loop, on the loop, at the boundary), how many agents compose the system, how deeply orchestration nests, at what scale the deployment runs, and what species of actor exercises each corrective function are all deployment variables. The tests bind over them without assuming any of them. What the tests fix is invariant across every configuration. Corrective chains terminate at the right party, with outward exercises terminating at the governed and answerability terminating at a sanctionable party, of which the human legal person is the current instance. Checking functions are structurally independent of what they check, in operator and in substrate. And every grant in every chain is a fresh act rather than an inference from persistence, including the grants under which the correction machinery itself operates. A framework that assumed the human's position, the agent count, or the exerciser's species would be falsified by the next shift in deployment practice. This one does not, because it binds termini, independence, and freshness rather than geometry.

3 Transparency Failure in Learned Systems

In deterministic infrastructure, disclosure of operative logic is sufficient to enable inspection. In learned systems, logic is distributed across architecture, parameters, training data, and representational schemas. Disclosure of source code alone does not reveal the operative decision boundary.

Principle 3.1 (Source Distribution)**.** Publishing inference code alone does not satisfy the CODE test. It allows an auditor to run the machine, but not to understand why it produces specific outputs.

To satisfy the CODE condition in learned systems, four components must be disclosed:

- **Logic (L)**: Model architecture and inference procedures
- **Weights (W)**: Learned parameters
- **Data (D)**: Training datasets with provenance
- **Representation (R)**: The operative categorical schema (ontological categories and label schemas)

We refer to this as **LWD-R transparency**. Omission of any component prevents meaningful external reconstruction of epistemic behavior.

To bridge this gap, compliance requires structured disclosure across the training pipeline, such as **Datasheets for Datasets** [Gebru et al., 2021] and **Model Cards** [Mitchell et al., 2019]. These artifacts function as the "source code" for structural evaluation; withholding them renders the system a black box.

3.1 The LWD-R Disclosure Requirement

For world models and agentic systems, the traditional transparency requirements (code, weights, data) are necessary but insufficient. Consider what it means to "inspect" a welfare eligibility model: seeing the inference code tells you how predictions are computed, but not *why* certain applicants are classified as high-risk. That "why" is encoded across four distinct layers.

Definition 3.1 (LWD-R: The Four Layers of AI Transparency)**.** A learned system satisfies LWD-R disclosure if the following artifacts are publicly available:

- **L (Logic)**: Model architecture and inference code (*how* predictions are computed)
- **W (Weights)**: Trained parameters (the learned patterns that produce outputs)
- **D (Data)**: Training corpus with provenance (*what* the model learned from)
- **R (Representation)**: The categorical schema (*how the deployed system sees the world*), in its operative form (Section 3.1.1)

Contestability Criterion. Transparency without contestability is surveillance. LWD-R disclosure is necessary but not sufficient: affected parties must have structural mechanisms to challenge representational categories, not merely observe them. A system publishing full LWD-R while providing no pathway for category contestation satisfies observability but fails constraint-layer corrigibility.

Why Four Layers? The first three (L, W, D) are familiar from "open source AI" discourse. The fourth, **Representation (R)**, is this framework's principal contribution to AI transparency discourse.

Why "R" Is the Novel Contribution. Existing "open AI" frameworks focus on L, W, and D. The Representation layer addresses a distinct failure mode: a model may publish weights, code, and data manifests while encoding *non-contestable categories* in its latent space. When an eligibility model classifies citizens as "high-risk," that category becomes an administrative fact. If the category itself cannot be challenged, the model has captured the state's interpretive sovereignty.

R is not transparency about *how* the model works; it is transparency about *how the model sees the world.* These categories (the model's ontology) are often undocumented and non-contestable: latent in nominal form, and taking their deployed shape through inference-time configuration (Section 3.1.1). A specific form of opacity arises when representational categories themselves become uncontestable ("ontological capture"), weakening constraint-layer contestability. When a model's classifications cannot be challenged by affected parties, the state has surrendered interpretive sovereignty regardless of weight publication.

Representation as Ontological Commons. The Representation layer is the layer at which AI infrastructure becomes a commons problem in the strict sense. Deterministic public infrastructure regulates access to a shared substrate — the registry, the rail, the payment system. Learned infrastructure regulates the substrate of meaning itself: the categorical schema through which an affected population can be administratively known. The categories under which citizens are classified as eligible or ineligible, risky or safe, deserving or undeserving constitute an **ontological commons** — a shared structure of perception that no single party can privately own without enclosing meaning itself. Because what the deployed system actually classifies depends on quantization, pruning, mixture-of-experts routing, and output filtering as much as on the trained weights (Section 3.1.1), the commons claim is over the *operative* schema, not the nominal one: a vendor who shifts meaning at inference through undisclosed deployment configuration has enclosed the commons as decisively as one who hides the underlying ontology, because the categorical structure that affected communities encounter has been moved beyond their reach. The contestability criterion stated above is therefore a commons-governance requirement in Ostrom's sense [Ostrom, 1990]: operative representational categories must remain subject to the polycentric, layered review that sustains shared resources, not delegated to a vendor whose latent space is by construction beyond user reach. The free-software anchor (CODE) supplies inspectability of the artifact; the commons anchor (R as ontological commons) supplies the governance condition under which inspectability is non-trivial; the cybernetic anchor supplies the closure condition under which contestation actually corrects.

R-Layer Change Control. The ontological-commons claim implies a velocity constraint. Operative representation drifts at retraining, fine-tuning, and filter-update cadence — days to weeks. A participatory ontology-review forum or

deliberative contestation process operates on a cadence of weeks to months. The Temporal Stability Condition of the companion paper [Aravind, 2026] states that a correction path slower than error accumulation is a runaway state (cf. Variety Drift, Section 4). Applied to the R layer: a schema that shifts faster than the affected community's capacity to contest it satisfies the form of contestability (a channel exists) while failing its function (the channel cannot close the loop before the schema has moved again). The Injunction Hook architecture (Section 6.9) shows how the framework handles machine-time/governance-time mismatches for actions; the same logic applies to representations.

Accordingly, operative representation changes above a declared materiality threshold in high-stakes EPI deployments are subject to the following change-control requirement: (i) **Pre-deployment notice** to the designated ontology-review body with a minimum review window (recommended: 21 days for material schema changes); (ii) **Suspensive effect** — objections from recognized affected-class representatives submitted within the review window impose a hold on deployment of the changed schema, pending resolution through a specified mechanism; (iii) **Versioned schema diffs** published in the LWD-R R-layer disclosure, so that drift is itself auditable: the auditor can reconstruct what the operative schema was on any given date and compare it against the categories a subject encountered. Where pre-deployment notice is operationally infeasible (emergency retraining to address safety failure), the change may proceed with immediate disclosure and a retroactive review window; the burden to demonstrate emergency falls on the operator. A schema that cannot hold still long enough to be contested fails GOVERN-over-R by the Temporal Stability Condition.

3.1.1 Operative Representation vs. Nominal Representation

Existing transparency frameworks frequently conflate a model's latent categorical structure at training time with its categorical structure at deployment. This framework distinguishes **nominal representation** (the latent space of the trained artifact in isolation) from **operative representation** (the categorical schema that emerges during inference under specific deployment conditions).

The distinction matters because deployment-time variables fundamentally modify the representational boundary, often without altering the disclosed weights [Hooker et al., 2019]:

- **Quantization** alters representational granularity; INT4 or INT8 inference collapses fine-grained categorical distinctions present in the FP16 reference checkpoint.
- **Pruning** removes edge-case distinctions; weights below a magnitude threshold are zeroed, eliminating rare but consequential decision pathways.
- **Mixture-of-Experts (MoE) routing** produces task-conditional ontologies; the "categories the model uses" depend on which experts the router activates for a given input, and routing tables are typically undisclosed.
- **Output filtering** (safety classifiers, refusal heads, regex post-processors) shapes which categories the user actually encounters in the deployed system, regardless of the underlying latent structure.

Two systems may share weights, training data, and inference code, and yet exhibit materially different representational behavior because deployment configurations differ. Disclosure must therefore describe *operative* representation, not *nominal* representation. Nominal transparency fails affected communities because the categorical structure they interact with in the deployed infrastructure differs from what the base model documentation suggests.

Proposition 3.1 (Operative-R Falsifiability)**.** Let R_{disc} be the disclosed representational schema and Σ the deployed system. The R requirement of LWD-R is satisfied if and only if an independent auditor, given the full disclosed LWD-R artifact set (logic, weights, data manifest), R_{disc}, and an enumeration of the deployment variables (quantization, pruning, routing, filtering, and the retrieval pipeline where present, per Principle 6.2), can reproduce the classification distribution of Σ on a published probe set within a declared statistical tolerance — e.g., total-variation distance at most ε at sample size n, with ε, n, and the probe set's provenance fixed in advance by the deployment tier. If the deployment variables are not enumerated in the disclosure, the derivation chain breaks and the system fails the R requirement.

The proposition converts "representational transparency" from a philosophical aspiration into an audit procedure. A vendor cannot claim R-disclosure by publishing the base-model architecture; the disclosure must extend to every component of the inference pipeline that selects, suppresses, or rewrites categorical outputs.

3.1.2 Synthetic Data and the Limits of Transferable Transparency

Transparency is not transferable through synthetic generation. A model trained on synthetic data generated by an opaque frontier model has not satisfied the Data (D) layer of LWD-R, regardless of how meticulously the resulting synthetic dataset is documented at the proximate step.

Documenting what is *in* a synthetic dataset is structurally distinct from accounting for how its content was shaped by the upstream model's biases, systemic omissions, and latent constraints. The synthetic-data manifest captures the *output distribution*; the disclosure obligation is over the *generative process*. The two are not equivalent: an opaque generator can produce a thoroughly documented synthetic corpus whose representational regularities still encode the generator's undisclosed inductive biases. Recent work on model collapse and the curse of recursive training reinforces this concern empirically: models trained on synthetic data inherit and amplify the systemic distortions of their generators [Shumailov et al., 2024].

When synthetic data trains a model that in turn generates synthetic data, accountability degrades recursively. Let the provenance chain be defined as

$$P = [g_0, g_1, \ldots, g_n],$$

where each g_i is the generator producing training data for g_{i+1}. The LWD-R disclosure obligation extends through the entire chain. Three regimes follow:

1. **Fully open chain.** Every g_i has full LWD-R available. The downstream model inherits transparent provenance; FORK and CODE are satisfiable in principle.
2. **Mixed chain with documented breakage.** At least one g_i is opaque, but the disclosure explicitly identifies the break-point and the residual uncertainty it introduces. The system fails strict LWD-R but supports bounded reasoning about its biases.
3. **Opaque-anchored chain.** The chain originates in or passes through a proprietary frontier model whose training data and process are undisclosed. Downstream documentation is performative; the structural opacity at g_0 propagates forward through every $g_{i>0}$ regardless of how thoroughly the proximate step is documented.

Principle 3.2 (Provenance Transitivity)**.** Transparency does not survive an opaque link. If any g_i in a provenance chain is structurally opaque, the entire downstream chain fails the D requirement of LWD-R, and the CODE condition of corrigibility fails for any system trained on its outputs.

This principle has direct procurement implications. A state agency that builds a "sovereign" welfare-eligibility model on synthetic data distilled from a proprietary frontier model has not achieved data sovereignty; it has inherited the upstream provider's representational biases under a documentary veneer. The opacity at g_0 is the binding constraint, not the visibility of g_1.

3.2 Strict Interpretability Firewall

The framework takes a strict position on what disclosure of operative representation requires for high-stakes deployments. We state the position, the reasoning behind its strictness, and the principal objection to it explicitly, because the alternative — a graduated certification regime — is the route by which structural transparency claims have historically been hollowed out.

Principle 3.3 (Strict Interpretability Firewall)**.** A learned system whose operative representation R cannot be disclosed at the standard required by Proposition 3.1 should not, consistent with this framework's legitimacy criteria, be adopted as Epistemic Public Infrastructure in the high-stakes tier (rights-affecting determinations: benefits eligibility, criminal-justice scoring, child-welfare classification, asylum adjudication). The failure is at CODE; no compensating provision in EXIT, AUDIT, GOVERN, or FORK substitutes for it.

The firewall's criterion is disclosure sufficient for independent behavioral reproduction of the operative representation (Proposition 3.1); it does not demand mechanistic interpretation of the latent geometry, which remains beyond inspection by design. The name marks where interpretability limits bite, namely adoption eligibility, not a claim that they can be solved.

The wording is deliberate. The principle does not assert that the law of any specific jurisdiction prohibits adoption; it states that adoption is inconsistent with the framework's structural criteria for legitimate public infrastructure. Whether jurisdictions choose to incorporate these criteria into binding administrative law is a separate question, addressed by procurement standards, statutory definitions of due process, and constitutional review.

Why strict, not graduated. A graduated regime would admit "provisional certification" for systems that meet partial R-disclosure (e.g., interpretability tooling without full operative-schema reproduction) on the condition of sunset clauses or remediation roadmaps. The structural prediction is that vendors optimize for indefinite renewal of provisional status. The same political-economy mechanism that converts "proportionate" contestation into sham governance (the Open-Loop Instability Argument, Section 1) converts "provisional" certification into permanent

partial compliance: the threshold becomes the floor, the remediation roadmap becomes the documentation of unmet obligations, and the sunset clause becomes the deadline that ratchets forward each renewal cycle. The graduated regime preserves the appearance of structural evaluation while removing its corrective force. The strict position accepts the cost of disqualifying current frontier deployments in high-stakes domains in order to preserve the diagnostic instrument.

The Human-Discretion Objection. The most serious objection is that human decision-makers — caseworkers, magistrates, social-services officers — are also non-interpretable in the strict sense, often biased, and frequently unaccountable, yet the framework does not propose to disqualify them from rights-affecting determinations. If the standard is non-interpretability, why does it bind machines and not humans?

Response: Bias-Plus-Corrigibility versus Bias-Minus-Corrigibility. The framework's claim is not that human discretion is unbiased. The claim is that human discretion is *corrigible* in ways current frontier learned systems structurally are not. A biased caseworker can be appealed against, retrained, removed, prosecuted, sued, or have their decisions individually reversed; their successor inherits institutional memory of why the reversal occurred; their replacement is a person of comparable capability available within the existing labor market. None of these conditions hold for a frontier model deployed at population scale: appeals against individual outputs do not propagate to the model's parameters, retraining is gated by compute the appellant cannot access, the operator cannot be "replaced" in the labor-market sense, and the model's behavior on a given input is reconstructable only by the operator. *Bias plus corrigibility beats bias minus corrigibility*, and the asymmetry is structural rather than rhetorical. The objection collapses two distinct properties — the presence of bias and the capacity for correction — and treats parity in the first as adequate to the comparison. The framework holds them apart and requires both.

Cost of the Strict Position. The strict position has costs that should be acknowledged rather than minimized. It implies that, as of the time of writing, every commercial frontier deployment in high-stakes EPI domains fails CODE under this framework. The framework is committed to that conclusion. The diagnostic instrument is only useful if it can name structural failure when the failure is widespread and economically inconvenient; an instrument calibrated to find current systems acceptable would not be a structural test, it would be a ratification procedure.

3.3 Machine-Verifiable Legitimacy

For learned and agentic systems, corrigibility must be paired with machine-verifiable legitimacy: (a) *authority encoding*—delegation and scope expressed as machine-interpretable credentials; (b) *revocation propagation*—immediate, observable revocation states that downstream execution environments respect; (c) *evidence standards*—tamper-evident execution traces and evaluation artifacts that meet defined admissibility criteria; (d) *incident-triggered mitigation*—automated mitigation gating and mandatory rollback protocols when thresholds are breached; and (e) *origination marking*—the record states whether a human or an agent operated the action, because agent operation of a human-gated ceremony produces artifacts indistinguishable from human operation. Absence of a marker is never evidence of human operation, so a deployment that relies on human operation must require a positive, signed marker. Consulting a marker may only narrow authority, never enlarge it. The unifying criterion is that authority and attribution are record-borne acts rather than inferences from artifacts. No authority from persistence. No operator species from ceremony. These are operational requirements; without them, architectural constraints remain unproven. Existing governance frameworks such as Abadie [2026] provide workflow orchestration but lack these machine-verifiable legitimacy requirements.

4 Variety Drift: Why AI Corrigibility Is a Persistence Condition

Traditional software remains static until explicitly patched. A tax calculation program written in 2020 will compute the same result in 2030, unless someone changes the code. Learned systems are different: they are *frozen snapshots* of a world that keeps moving.

The Problem. An AI model trained on 2024 data encodes 2024 patterns. By 2026, the world has changed: new fraud techniques emerge, economic conditions shift, language evolves. The model's "variety" (its capacity to handle diverse situations) was fixed at training. The world's variety keeps expanding. The gap between them grows over time.

Definition 4.1 (Variety Drift)**.** Let $V_{\text{world}}(t)$ be the variety of situations the environment presents at time t, and $V_{\text{model}}(\theta)$ be the variety the model can handle, fixed at training. **Variety Drift** is the growing gap:

$$\Delta(t) = V_{\text{world}}(t) - V_{\text{model}}(\theta) \tag{1}$$

Why This Matters. Without mechanisms for retraining accessible to the community, variety drift widens indefinitely (Figure 2). A model that passed all five corrigibility tests at deployment may fail them two years later. Not because anyone changed anything, but because the world moved and the model did not. **AI corrigibility is not a deployment certification; it is a persistence condition.**

The Gap Closes from Both Ends. Variety Drift is the model-side half of the temporal problem. The controller side erodes too: skills exercised only through a system atrophy outside it, the long-documented irony of automation [Bainbridge, 1983], so the governance variety available for correction decays endogenously even as environmental variety grows. The persistence condition is therefore conservative. Measuring controller-side erosion alongside model-side drift is an open problem.

The Revalidation Threshold. When variety drift exceeds a domain-specific threshold τ_{drift}, mandatory revalidation is triggered:

$$\Delta(t) > \tau_{\text{drift}} \implies \text{Revalidation Required} \tag{2}$$

The threshold τ_{drift} is not universal; it depends on consequence severity. High-stakes domains (welfare eligibility, criminal risk assessment) require lower thresholds than low-stakes domains (content recommendation). Setting τ_{drift} is a governance decision, not a technical one.

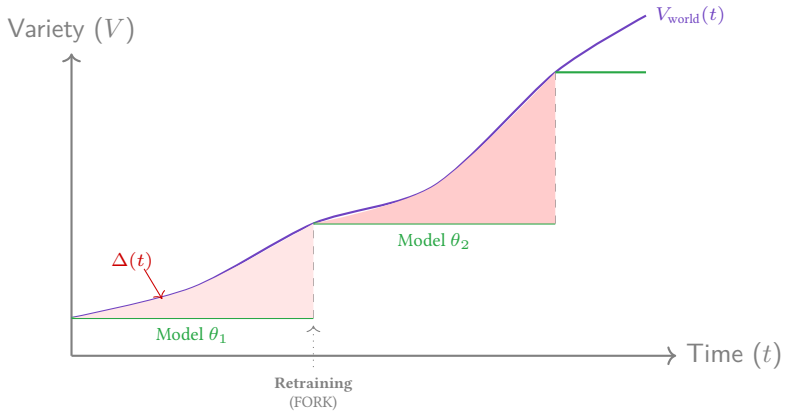

Figure 2: **The Temporal Variety Gap.** Environmental variety $V_{\text{world}}(t)$ evolves continuously (curve). Model variety $V_{\text{model}}(\theta)$ holds still between retraining steps (stepped line). The shaded area is the accumulated variety gap $\Delta(t)$. Only **FORK** rights allow the gap to be closed via retraining.

Claim 4.1. A learned system that passes all five corrigibility tests at deployment may fail them over time if the community lacks the resources to retrain. Corrigibility for AI is a dynamic property, not a static certification.

5 Resource Barriers to Forkability

In traditional software, forking is cheap: copy the code, set up a server, deploy. The cost is measured in storage and labor. In AI, forking is expensive: the cost is measured in *compute*, the GPU-hours required to retrain a model from scratch.

The Structural Difference. When Elastic changed its license, AWS forked Elasticsearch and the community continued development. This was possible because the resource barrier was low. When Meta releases Llama weights, can the community "fork" Llama in the same sense? They can *run* it (inference). They can *adapt* it (fine-tuning). But can they *reproduce* it from scratch if Meta's training data turns out to be problematic? For frontier models, the answer is no: the compute cost exceeds what any non-corporate actor can access. Table 4 stratifies the costs across reproduction layers.

Claim 5.1. Legal permission to fork without access to compute or data is meaningless. This is an infrastructure question, not a licensing question.

5.1 Compute Capture

The Problem. A company releases model weights under an open license. Headlines announce "open source AI." But the training run consumed compute at frontier scale. No academic consortium has to date independently reproduced

Layer	Cost	Barrier
Inference code	Minimal	None
Running open weights	Low	Minor
Fine-tuning	Moderate	Economic
Retraining from scratch	Very high	Structural

Table 4: Resource barriers to forking learned systems

a frontier training run, and public research compute is not provisioned for routine reproduction — which is precisely the gap the Public Compute Nodes proposal below addresses. If the model exhibits problematic behavior traceable to training data, the community can observe the symptoms but cannot fix the cause. They can run the model; they cannot reproduce it.

Definition 5.1 (Compute Capture)**. Compute Capture** occurs when the compute cost to retrain a model (C_{train}) exceeds the compute accessible to non-operator actors ($C_{\text{accessible}}$) by more than a policy-determined threshold:

$$C_{\text{train}} > \kappa \cdot C_{\text{accessible}} \tag{3}$$

where κ is an accessibility multiplier (typically $1 \leq \kappa \leq 2$). Under Compute Capture, releasing weights provides **inference forkability** (you can run it) without **training forkability** (you can reproduce it).

Why This Matters. If training cost exceeds $100\times$ global academic compute capacity, the FORK test fails regardless of licensing terms. Compute asymmetry is epistemic stratification: when only the most-resourced operators can execute a training fork, the capacity to correct an algorithmic worldview is structurally withheld from the populations most exposed to its errors.

The formal FORK determination for training becomes:

$$\text{FORK}_{\text{training}} = \mathbf{1}[C_{\text{train}} \leq \kappa \cdot C_{\text{accessible}}] \wedge \mathbf{1}[\text{data\_manifest\_available}] \tag{4}$$

where $\mathbf{1}[\cdot]$ is the indicator function.

5.2 Data Capture

Training forkability requires more than compute access; it requires data sovereignty. Compute Capture and Data Capture are independent failure axes for training forkability: a system can satisfy one and fail the other and still be unforkable in practice. Discussions of "open AI" that focus exclusively on compute miss this second axis.

Definition 5.2 (Data Capture)**. Data Capture** occurs when the data required to train a functionally equivalent model can only be produced by infrastructure the open community cannot reproduce.

This barrier extends well beyond raw text corpora into a taxonomy of upstream training dependencies:

- **Synthetic datasets** generated by proprietary frontier models, which inherit the generator's distributional biases (cf. Section 3.1.2).
- **Distilled expert outputs** produced by querying closed APIs at industrial scale; the resulting corpus is legally encumbered and practically irreproducible without paid API access.
- **Proprietary evaluation sets** that anchor benchmark results; without access, third-party reproductions cannot demonstrate functional equivalence.
- **RLHF reasoning traces and preference data** collected through proprietary annotator pipelines whose annotator demographics, instructions, and quality gates remain undisclosed.
- **Reward models** trained on those preferences; downstream RL fine-tuning cannot be replicated without the reward model's weights and training data.
- **Tokenizer and pre-training filter rules** that silently shape what content survives into training.

Formally, Data Capture occurs when the data corpus required for reproduction (D_{required}) exceeds the data accessible to non-operator actors ($D_{\text{accessible}}$) beyond a policy-determined friction threshold:

$$D_{\text{required}} > \kappa_D \cdot D_{\text{accessible}} \tag{5}$$

where $\kappa_D \geq 1$ is a data-accessibility multiplier analogous to the compute multiplier κ in Definition 5.1. The threshold accounts for non-trivial substitutes: a public corpus that is empirically equivalent to the proprietary one (e.g., Common Crawl filtered to match a vendor's quality gates) may close the gap, but only if the equivalence is auditable.

Proposition 5.1 (Joint Forkability Condition)**.** Training forkability is genuinely available only if the required artifacts (Table 5) are available and the system evades both Compute Capture and Data Capture simultaneously; the resource conjunction strengthens (and supersedes the manifest clause of) the earlier $\text{FORK}_{\text{training}}$ condition:

$$\text{FORK}^{\text{full}}_{\text{training}} \;=\; \mathbf{1}[C_{\text{train}} \le \kappa \cdot C_{\text{accessible}}] \;\wedge\; \mathbf{1}[D_{\text{required}} \le \kappa_D \cdot D_{\text{accessible}}].$$

The proposition reframes the open-weights debate. A frontier model released under a permissive weight license but trained on a proprietary RLHF corpus and a non-public reward model has not crossed the FORK threshold; it has shifted the capture vector from compute to data. The community can run the model and fine-tune it cheaply, but it cannot independently produce the next checkpoint, audit the alignment behavior, or replace the operator. Data Capture is therefore the structural sibling of Compute Capture, and credible LWD-R disclosure must address both.

Public-Compute Insufficiency. A state policy that funds public compute infrastructure without simultaneously funding public data infrastructure (open corpora, open evaluation sets, open RLHF preference pools, open reward-model training) will produce inference forkability at best. The state will be able to *run* models but not *correct* them. Genuine training forkability is a two-track infrastructure investment, not a single-track one.

5.3 Compute as Governance Infrastructure

This analysis implies that **compute is itself DPI**: a common-pool resource requiring multi-stakeholder governance rather than unilateral corporate control. Legal forkability is insufficient absent practical compute accessibility. Architectural responses include:

1. **Public Compute Nodes**: State-sponsored clusters with multi-stakeholder governance
2. **Mandatory Cost Disclosure**: FLOP estimates and data manifests
3. **Federated Training Pools**: Distributed training across independent nodes

5.4 Minimum Evidence for Learned System Forkability

To prevent "FORK fails" from becoming a rhetorical claim rather than a verifiable determination, learned systems must satisfy specific evidentiary requirements:

Table 5: Evidence requirements for FORK in learned systems

Artifact	Status	Verification
Inference code	Required	Open source license; build reproducibility
Model weights	Required	Public download; hash verification
Training data manifest	Required	Datasheets [Gebru et al., 2021] with provenance
Training checkpoint	Recommended	Enables fine-tuning without full retrain
Training code	Required	Includes preprocessing, hyperparameters
Compute cost estimate	Required	FLOP estimate or cloud cost documentation
Successful reproduction	Decisive	Third-party retrain within 10% of benchmark

Forkability Determination. A learned system passes FORK if:

1. All "Required" artifacts are publicly available under permissive terms
2. The compute condition holds: $C_{\text{train}} \le \kappa \cdot C_{\text{accessible}}$ for non-operator actors
3. The data condition holds: the manifested data, or an auditably equivalent substitute, is accessible within κ_D (Definition 5.2)
4. Either a successful third-party reproduction exists, OR the training code + data manifest enables reproduction in principle

Artifact-Reproducible vs. Governance-Reproducible: The Operational-Capacity Layer. The evidence table above addresses artifact forkability — whether the artifacts required to retrain or reproduce the system are available. It does not address operational-capacity forkability: whether the forking community possesses the tacit knowledge, evaluation infrastructure, and operational lore required to *govern* the forked system after reproducing it. A fork that copies the source and leaves the tacit operational knowledge behind — the evaluation harnesses, red-team suites,

known-failure registries, deployment intuition, and incident history accumulated through production operation — has reproduced the artifact without reproducing the corrective apparatus. A governance-incapable fork is corrigible in name only: it reproduces the system's behavior but not the community's ability to detect and correct its failure modes.

This gap is the epistemic twin of Compute Capture. Where Compute Capture prices the economic barrier to training forkability, operational-capacity deficit prices the knowledge barrier to *governance* forkability. The Edge/SLM fragmentation finding (Section 8) illustrates its other face: distributing the artifact without the coordination capacity distributes incorrigibility, not correction.

FORK evidence accordingly distinguishes two levels:

Artifact-reproducible The artifacts in Table 5 are available and a third party can retrain within benchmark tolerance. This is the current standard.

Governance-reproducible The above, plus: evaluation suites (functional test harnesses covering the deployment domain), red-team scenarios and known failure registries, incident logs from production operation (sanitized for privacy), and harness documentation sufficient for an independent team to operate and assess the system without access to the original operator. Governance reproducibility is the FORK standard for `high_stakes` tier deployments.

Audit artifacts for `high_stakes` tier assessments must record which of the two levels is established — a `fork_viability` determination; the corresponding audit-artifact schemas are maintained in the schema repository (`https://github.com/anivar/corrigibility-schema`).

5.5 Inference vs. Training Forkability

Systems that publish weights but withhold training data or code achieve only **inference forkability**, not **training forkability**. Inference forkability permits running the model; training forkability permits correcting it. Only the latter satisfies FORK for governance purposes.

Concrete Examples. As of mid-2026, the open-model landscape divides cleanly along this axis:

- **Inference forkability only:** Llama 3.1–4 (Meta), DeepSeek-V3 and DeepSeek-R1, Gemma 2–3 (Google), Mistral Large. Weights are published under varying license terms, but training data is not released, full preprocessing pipelines are not disclosed, and reinforcement-learning artifacts (reward models, RLHF traces, distillation corpora) remain proprietary. Users can run and fine-tune these models but cannot independently reproduce or audit the training process.
- **Training forkability:** the OLMo family [Groeneveld et al., 2024] (Allen Institute for AI; OLMo 3 extends the release to the full model-flow artifact chain) and Pythia [Biderman et al., 2023] (EleutherAI). Both publish weights, complete training code, full data manifests (the OLMo training mixtures and The Pile [Gao et al., 2020] respectively — the later takedown of one Pile component is itself an instance of data-layer fragility in FORK determinations), intermediate checkpoints, and training logs sufficient for third-party reproduction. These projects, alongside their data releases, satisfy the FORK test in principle and (for partial reruns) in practice.

The distinction matters: when Llama, Gemma, or DeepSeek exhibit unexpected behavior, the community can observe symptoms but cannot diagnose root causes in training data. When an OLMo-family model exhibits unexpected behavior, researchers can trace the issue to specific training examples. **Open weights with closed training pipelines is inference transparency without correction capacity.**

Systems claiming "sovereign" or "indigenous" status through fine-tuning foreign base models achieve inference forkability only; the base model remains opaque and unmodifiable. Such claims are structurally hollow: the fine-tuned layer is correctable, but the foundational model on which it depends is not.

Alignment with OSI Definition. The Open Source Initiative's Open Source AI Definition 1.0 [Open Source Initiative, 2024] codifies this distinction, requiring: (1) data information sufficient for recreation, (2) complete training code, and (3) model parameters. By this standard, the Llama, Gemma, DeepSeek, and Mistral release families surveyed above do not qualify as open source AI despite marketing claims; the OLMo family and Pythia do. The EU AI Act [European Parliament and Council of the European Union, 2024] similarly distinguishes between models with varying transparency obligations. This framework's FORK test operationalizes the same principle: without training forkability, governance correction is structurally impossible.

6 Agentic Scaling and Governance Denial of Service

The preceding sections establish that learned systems face unique barriers to corrigibility: opaque training pipelines (violating CODE), frozen variety (the Temporal Variety Gap), and compute-gated reproduction (violating FORK). These challenges intensify as AI moves from isolated models to autonomous agents operating within infrastructure. As DPI transitions to support the "Agentic Economy," corrigibility moves from a moral preference to an engineering requirement. Autonomous agents lack the biological resilience to handle bureaucracy.

Principle 6.1 (Structural vs. Alignment Distinction)**.** This framework distinguishes structural corrigibility from "AI Alignment." The AI safety literature has extensively studied alignment (ensuring systems pursue operator-intended goals) [Soares and Fallenstein, 2014] and the "off-switch" problem [Hadfield-Menell et al., 2017]. These framings center on *operator control*. The safety literature also uses *corrigibility* itself for an operator-facing property: an agent's disposition to accept correction and shutdown from its principal [Soares et al., 2015]. The two usages are complementary, not competing; this paper's is structural and subject-side. Alignment asks: *Can the operator control the system?* Corrigibility, as used here, asks: *Can the affected subject correct the system?* Structural legitimacy requires the latter.

6.1 Agent Systems as Composable Architectures

An agent system is not a model; it is a composable architecture. Treating "the AI" as a synonym for "the model weights" obscures the components that actually determine administrative behavior. Formally, an agent system A is the tuple

$$A = (M, H, B, S, T, C, \bar{M}),$$

where:

- M is the underlying **model** (weights and inference logic);
- H is the **orchestration harness** (planner, scheduler, control flow, retry policy);
- B is the **action boundary** (the deterministic validation envelope of Section 6.7);
- S is the set of **action specifications** (the declarative descriptions of what actions the agent may invoke);
- T is the set of **tool definitions** (the concrete executables, APIs, or functions that actions resolve to);
- C is the **deployment-time configuration** (system prompts, retrieval pipelines, output filters, environment variables);
- $\bar{M}$ is the **persistent memory layer** (per-subject context arrays, conversation history, cross-session retention, vector-store residues), distinct from the model parameters M in that it accumulates over time as a function of the deployment's interaction history.

Each component bears distinct governance requirements, and each can capture or release sovereignty independently of the others. Behavior is a property of the system A, not the model M. The same model M deployed in a different harness H, with different action specifications S, against different tool definitions T, under different configuration C, with different memory state $\bar{M}$, produces materially different administrative outcomes. Two state agencies running the same open-weights model may yield incompatible welfare-eligibility decisions because H, S, T, C, or $\bar{M}$ differ.

Why System Behavior Is Not Model Behavior. Because H, S, T, C, and $\bar{M}$ modify behavior independent of M, model-level LWD-R disclosure is necessary but insufficient for agentic infrastructure. If a state deploys an open-weights model but orchestrates its actions through a proprietary vendor harness, the infrastructure fails both CODE (the orchestration logic is closed) and FORK (the vendor harness is not reproducible). Conversely, a fully open harness wrapped around a closed model leaves the inference layer opaque but makes the action boundary auditable; the action-boundary *component* is independently verifiable, while the system as a whole still fails CODE at the model layer and, under the strict firewall (Section 3.2), remains ineligible for high-stakes deployment.

Memory as Liability. The persistent memory layer $\bar{M}$ is structurally distinct from the other components in that its content is a running function of subjects' own interaction histories rather than a deployment-time decision of the operator. This makes $\bar{M}$ both an attack surface (cross-subject contamination, prompt-injection persistence across sessions, latent profile-building beyond the disclosed retention policy) and a rights surface (data-protection regimes such as GDPR's right-to-erasure attach directly to its contents, irrespective of how the operator has classified the deployment). Public agentic infrastructure must therefore expose a programmatic EXIT for $\bar{M}$: a deterministic, auditable interface that purges per-subject context arrays on request, on schedule, and on revocation, with the wipe verifiable against the same evidence standards as the audit log itself. "We will not retain it next time" is not an EXIT for memory; deletion that the subject cannot independently confirm is performative compliance, not corrigibility.

Claim 6.1. Disclosure obligations attach to the agent system A, not to the model M alone. "Open weights" is a property of M; corrigibility is a property of A.

Nested Delegation. The tuple is recursive. The harness H may itself contain agent systems A_i, as when a planning agent spawns worker agents that spawn verifier agents, or when an institution's harness wraps a vendor's harness that wraps a model's own tool loop. The tests then compose across delegation depth. Each inner system's authority must be contained within its parent's (attenuation), audit records must reconstruct across nesting levels, EXIT must propagate down the chain (the Workflow EXIT layer below), and the outer validator cannot see the inner systems' representations, so R-opacity nests. The weakest-link principle extends naturally. The corrigibility of a nested deployment is the minimum over its delegation depth, and enforcement placement is what makes depth tractable: because the action boundary attaches at the effect surface rather than per model, the number and nesting of internal agents is invisible to the validator. A workflow of a hundred subagents is, to the boundary, one system emitting proposals.

Component-Level Disclosure Map. Each component of A has a distinct disclosure surface and a distinct failure mode under capture (Table 6). The five corrigibility tests apply componentwise: a system that publishes weights (M) but hides the harness (H), action specifications (S), and configuration (C) satisfies CODE only at the model layer. Affected parties cannot audit, contest, or fork the components that actually mediate their interaction with the state.

Table 6: Agent-system components and their corrigibility surface

Component	Primary tests	Capture failure mode
M (model)	CODE, FORK	Compute Capture, Data Capture, opaque RLHF
H (harness)	CODE, FORK	Proprietary planners, closed orchestration
B (action boundary)	GOVERN, AUDIT	Validator bypassable via prompt injection
S (action specifications)	CODE, GOVERN	Closed schemas; action ontology left implicit in the model
T (tool definitions)	AUDIT, FORK	Closed APIs; vendor-locked side effects
C (configuration)	CODE, AUDIT	Hidden system prompts, undisclosed retrieval
$\bar{M}$ (persistent memory)	EXIT, AUDIT	Cross-subject contamination, undisclosed retention, deletion that the subject cannot verify

6.2 Retrieval Opacity

A particularly common deployment pattern places retrieval-augmented generation (RAG) inside the configuration layer C. The model retrieves documents from a state-controlled corpus and conditions its outputs on the retrieved text. Vendors often describe such systems as "open" on the basis of disclosed weights, even when the retrieval index, ranking function, embedding model, and corpus are proprietary.

Principle 6.2 (Retrieval Opacity)**.** Retrieval-augmented systems are not structurally open if their retrieval components are opaque, even when the underlying model M is fully disclosed. The retrieval pipeline is part of the operative representation R (Section 3.1.1); concealing it concedes the categorical schema to the operator.

The principle has two consequences. First, the disclosure surface for C must include the retrieval corpus manifest, the embedding model and its training data, the ranking function, and the retrieval logs at evaluation time. Second, "RAG with open weights" is not a corrigibility claim; it is at best a partial CODE claim. The R layer of LWD-R remains undischarged because the categorical structure that conditions every inference is sourced from an undisclosed index.

6.3 Corrigibility as an API Contract for Agents

- **EXIT is Exception Handling**: To an agent, a mandatory system with no exit path appears as a logic deadlock.
- **FORK is Failover**: A dependency that cannot be substituted is a single point of failure no resilient workflow accepts.
- **CODE is Documentation**: Agents require inspectability of logic to form valid plans.
- **AUDIT is Observability**: An optimization agent cannot function without a reliable failure signal.
- **GOVERN is Constraint Discovery**: Advisory guidelines result in undefined behavior for software agents.

Claim 6.2. Incorrigible infrastructure is structurally incompatible with AI automation.

6.4 The Human Friction Buffer

Incorrigible infrastructures may appear stable under human load because humans absorb friction through delay, compliance, and informal workaround. Autonomous agents remove this buffering layer.

Illustrative Scenario. Consider an autonomous procurement agent that interacts with a mandatory payment API. On encountering a compliance flag, the agent receives an opaque error code. No machine-readable error taxonomy exists, and dispute resolution is human-mediated. The agent retries, triggering repeated failures and queue saturation. Human agents absorb such friction via manual override and phone calls; fleets of autonomous agents cannot. The result is a governance denial of service (GDoS) that cascades across the payment rail.

Governance Denial of Service. Under agentic scaling, governance capacity can be saturated ("governance denial of service"). When autonomous agents encounter errors, they retry programmatically rather than waiting patiently; and because one instruction can legitimately spawn many agents, governance load scales with orchestration fan-out, not with the number of principals. Governance mechanisms designed for human patience saturate under machine speed. This is a constraint-layer failure that the Action Boundary must mitigate.

Exception Handling at the Margin. The friction being removed was never only delay. In bureaucratic systems, informal human discretion functions as undocumented variety absorption. The clerk processes the case whose lived reality fails the literal text of the rule, the applicant with a non-standard household, a disputed record, an unreadable fingerprint. Agentic workflows executed over deterministic interfaces remove that absorption without replacing it. Agents optimize for legible, clean pathways, so the users exhibiting the highest environmental variety, who are precisely the least-resourced stratum of the gradient quantifier, meet immediate and repeatable exclusion where a human intermediary once absorbed the mismatch. Removing the friction buffer without explicit exception pathways and guaranteed fallbacks in the action specification concentrates the automation's failure mode on the population least equipped to contest it. The correction machinery of Section 6.9 exists for exactly this case.

6.5 Agentic Interoperability Requirements

For an autonomous agent $\mathcal{A}$ to operate reliably within infrastructure $\mathcal{I}$, the following predicates must be strictly True:

1. **Reversibility:** $\forall$ state $s_t \in \mathcal{S}$, $\exists$ transition s_{t+1} (via **EXIT**) such that $\text{Cost}(s_{t+1}) < \infty$. (No deadlocks).
2. **Decidability** (constraint discoverability, not decidability in the computability sense)**:** $\forall$ input x, $\text{Compute}(x) \to \{0, 1\}$ is inspectable via **CODE**. (No hidden constraints).

Failure of these predicates renders $\mathcal{I}$ a "Hazmat Environment" for automated agents, effectively blocking the agentic economy.

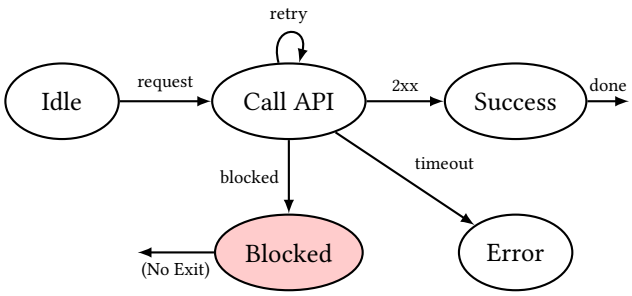

Figure 3: **Agentic automaton.** Without EXIT and observable error codes, agent loops indefinitely, times out, or escalates incorrectly.

6.6 EXIT as the Architecturally Privileged Test in Agentic Systems

Across the five corrigibility tests, EXIT acquires structural primacy in agentic deployments that it does not have in deterministic infrastructure. Three properties of learned and agentic systems—*stochasticity* (outputs cannot be guaranteed by inspection), *persistence* (the memory layer $\bar{M}$ accumulates state outside any single transaction), and *autonomy* (the system initiates state changes without human action-by-action consent)—each independently raise the cost of a missing EXIT. Together they make EXIT the test whose failure most reliably converts a learned-systems deployment from infrastructure into capture.

The Five EXIT Layers in Agentic Systems. A deployment that nominally provides "opt-out" typically provides one of the following while leaving the others unguarded; corrigibility under agentic conditions requires all five.

1. **Subject EXIT.** Affected parties can refuse an AI-mediated determination and obtain a non-AI alternative or a human-authored decision without disproportionate penalty (per the EXIT row of Table 3). This is the canonical EXIT inherited from the deterministic case.
2. **Memory EXIT.** The persistent memory layer $\bar{M}$ exposes a programmatic, auditable purge interface (Section 6.1, "Memory as Liability"). Without this, "opting out" leaves a residue the subject cannot verify is gone, and data-protection regimes (GDPR's right-to-erasure, India's DPDP Act §12) attach to that residue regardless of operator classification. Purge alone is not the test. Termination counts only if resumption requires a fresh grant, so nothing about the ended authority may be inferable from what persists.
3. **Workflow EXIT.** Halt signals propagate deterministically through agent chains: when a parent agent invokes a child agent or a tool, the architecture must define how cancellation, timeout, and termination flow back through the chain. Without this, autonomous chains exhibit the deadlock pathology of Figure 3 at fleet scale, producing the Governance Denial of Service described above. Propagation is also not the whole test. A mandate that expires cleanly and is then re-instantiated from residual context has not exited, whether the residue is a still-running orchestration graph, prior approvals in memory, or a scheduler firing the next run. EXIT is satisfied only where authority cannot regenerate from persistence.
4. **Operator EXIT.** The operator retains a deterministic shutdown and pause capability for the deployment, including under partial failure. The AI-safety literature treats this as the "off-switch" problem [Hadfield-Menell et al., 2017]; corrigibility analysis adopts the same engineering requirement but assigns it a different normative weight. Operator EXIT is necessary but not sufficient: a system corrigible only to its operator is not corrigible to those it governs.
5. **Bystander EXIT.** Parties an agent's actions land on who hold no addressable position in its operator-subject loop require structural recourse independent of that loop. The class has two structurally distinct members. The unrepresented third party stands outside the delegation entirely: the counterparty to an autonomous procurement agent, the recipient of an automated communication, the person whose data is incidentally processed. The represented party stands inside the delegation as its principal but outside its record: a ward whose guardian operates the wallet that answers for them is acted for under mandate, not acted upon from outside, yet holds no independent addressability in the artifacts the mandate produces. The two positions take different corrective instruments: the third party needs standing against a relationship they were never in; the represented party needs a record channel inside a relationship conducted in their name. The juridical-mechanism families of Section 6.9, particularly Injunction Hooks (Figure 5) and class-action vehicles, are the legitimate sites for both; their absence converts agent autonomy into a unilateral expansion of operator reach.

Why EXIT Is Harder for Agents Than for Deterministic Systems. In deterministic infrastructure, refusal is a single-point property: the subject either participates or does not. In agentic deployments, refusal must be specified along a temporal axis (the agent has already acted, and may continue to act), a memory axis ($\bar{M}$ retains traces of refused interactions), a chain axis (downstream agents may have inherited state), and a counterparty axis (other parties may be mid-interaction with the agent). A deployment that provides only the first dimension—a button labelled "stop"—is structurally indistinguishable from a deployment that provides nothing, because the system can re-enter the subject's life through any of the other four channels.

Claim 6.3. A learned or agentic deployment failing any of the five EXIT layers cannot satisfy the EXIT test in this framework, irrespective of compliance with CODE, AUDIT, GOVERN, or FORK. The irreducibility argument of Section 2 therefore tightens under agentic conditions: where deterministic infrastructure can fail one EXIT dimension and carry PARTIAL as a diagnostic annotation (the determination itself still resolves PARTIAL to FAIL), agentic infrastructure cannot: failing any layer is total.

6.7 The Action Boundary Protocol

As infrastructure transitions to the Rule of the Workflow, the fundamental unit of governance can no longer be the static database record, nor can it be the stochastic model weight. To subject autonomous systems to democratic constraint, the architecture must adopt a new structural primitive: the **Action Boundary**.

Principle 6.3 (Action Boundary Protocol)**.** If the state cannot govern the neural network's weights, it must govern the **action boundary**: the deterministic envelope that wraps stochastic inference. The model proposes; the validator disposes. The Action Boundary Protocol's five architectural requirements are specified below.

In agentic systems, learned inference should not directly trigger irreversible action. Instead, inference outputs must pass through a deterministic validation layer that enforces policy constraints and safety rules.

Architectural Proposals and the Legitimacy Gap. Recent architectural proposals such as the DPI-AI Framework [Abadie, 2026] advance composable AI capabilities ("AI Blocks") orchestrated through "DPI Workflows." This framing correctly treats AI as modular infrastructure rather than institutional magic. However, composability of *capability* must be matched by composability of *legitimacy*. If AI Blocks are callable, authority must also be callable, bounded, signed, and revocable. If workflows orchestrate identity, policy, and inference, then risk must be tiered explicitly: informational assistance is not equivalent to a benefits denial; conditional automation is not equivalent to a rights-affecting determination. The Action Boundary Protocol provides this structural requirement: probabilistic inference separated from deterministic execution through enforceable validation layers.

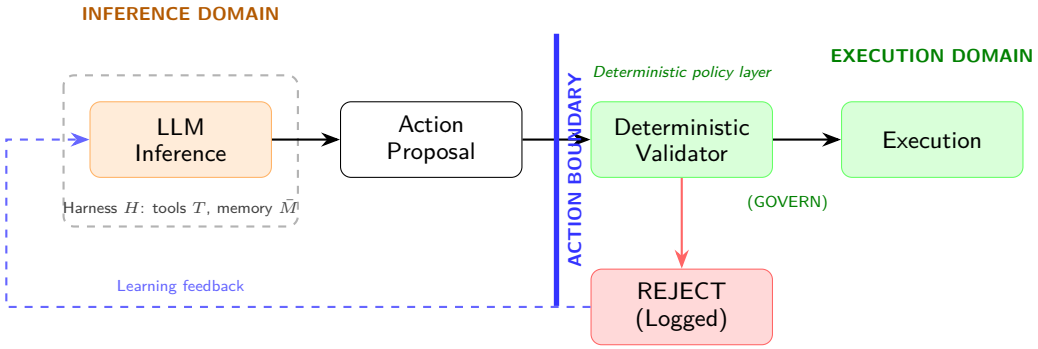

Figure 4: **Action Boundary Protocol.** Probabilistic inference (orange) produces action proposals from within its harness envelope (orchestration, tools T, persistent memory $\bar{M}$): internal delegation, however deep or wide, is invisible to the validator, which receives only typed proposals. The **deterministic policy layer** (green, implementing GOVERN) enforces hard constraints before execution. Invalid actions are rejected, logged, and fed back for learning. The vertical **Action Boundary** separates the inference domain from the execution domain: the architectural instantiation of corrigibility in agentic systems.

Figure 4 illustrates the architecture: the validator enforces hard constraints independent of probabilistic inference. Corrigibility in agentic systems therefore requires strict separation between epistemic generation and action authorization.

Architectural Requirements. The Action Boundary Protocol specifies five architectural requirements for agentic interoperability. Multiple open protocols can instantiate these requirements; the structural necessity is independence from any specific external standard.

1. **Deterministic Validation:** Inference outputs pass through a hard-coded policy layer before execution. The validator computes its decision through ordinary program logic and is independent of the model's probabilistic computation.
2. **Context-Window Isolation:** Constraints are encoded in the execution environment, not injected into the model's context window. A learned system cannot prompt-inject past constraints it cannot read; isolation is the structural answer to injection that attempts to cross the policy boundary. Injection that steers the model toward harmful-but-in-policy proposals remains, and must be bounded by narrowing action classes and tightening the validator's predicates.
3. **Exception Handling (EXIT):** The action specification dictates explicit fallback paths and termination triggers, preventing the infinite loops that produce Governance Denial of Service (GDoS).
4. **Machine-Readable Specifications (CODE & AUDIT):** The agent's workflow and tool permissions are defined in declarative, independently auditable schemas. The system's operational ontology becomes public, discharging the workflow layer's share of the Representation (R) requirement of LWD-R, and execution emits structured transaction logs satisfying AUDIT.
5. **Binding Constraints (GOVERN):** The validator acts as a non-overrideable controller. Actions that fail the schema are rejected and logged, providing an auditable failure signal rather than silent compliance.

Subject Receipt Requirement. The action boundary specifies what the validator enforces on the operator's behalf. Structural corrigibility also requires that the individual subject of each determination receive a cryptographically

verifiable record of it. Without such a record, AUDIT verifies the *system-as-logged* against the *operator's specification* but nothing verifies the *log* against the *lived event*: the subject cannot confirm that the determination they experienced is faithfully recorded, and auditors replaying signed logs have no ground-truth check against the population's actual encounter with the system.

The action boundary must therefore emit, for each rights-affecting determination, a **subject receipt**: a signed, structured record including the action class, the validator decision, the timestamp, and an inclusion proof against the audit log (transparency-log style, per RFC 9162 [Laurie et al., 2021] or equivalent). The receipt must be: (i) **subject-facing** — delivered to the individual through a channel independent of the operator's logging infrastructure, accessible offline; (ii) **independently verifiable** — the inclusion proof must allow the subject, or any party they designate, to confirm that the receipt corresponds to an entry in the canonical audit log; and (iii) **machine-readable** — in a format that class-action aggregators and juridical mechanisms can ingest as evidence, because the distributed/class-action and juridical mechanism families (Section 6.9) depend on these receipts as the primary evidence that a systemic pattern exists.

The emission obligation attaches to the determination, not to the boundary. Deterministic infrastructure, which has no action boundary, emits the same receipt at the effect surface where the determination becomes a state change in the subject's record; the construct is therefore available to the companion framework's audit rules [Aravind, 2026] without presupposing an agentic substrate.

The receipt object itself has close relatives in the emerging agent-infrastructure literature: receiver-attested proposals anchor per-action records, signed by the receiving service, in witnessed transparency logs with inclusion proofs, and adjacent designs add offline verification by parties who need not trust the issuer. What none of the designs surveyed supplies is the addressee. Each addresses its receipts to the operator, the relying party, or the auditor; the person the determination lands on appears nowhere in the protocol. The subject receipt differs there, solely and decisively: it is emitted to that person, over a channel the operator does not control, verifiable and aggregable by them or their designee.

For high-stakes tiers, the system must additionally expose a **citizen-constraint registry**: a channel through which recognized subject-class representatives can lodge protective constraints — specific categories of determination that a class certifies as producing systematic harm — that the validator must evaluate before executing determinations in that class. This is the citizen-side counterpart of the recognized court's halt flag in the Injunction Hook: it gives the governed structural standing at the boundary they are governed by, not only recourse after the fact.

Why Outside-Context Enforcement Is Categorically Different. The five requirements share a common architectural commitment: policy is enforced *outside* the inference context, not within it. This is not a quantitative improvement ("stronger guardrails") but a categorical one ("a different kind of guarantee"). In-context guardrails — system prompts, instruction-following fine-tuning, output classifiers conditioned on the same model — share the inference layer with the content being judged. Any string the model can read is a string the model can be persuaded to ignore, reframe, or override. The guardrail and the input occupy the same substrate; the boundary between policy and payload is a probabilistic prior, not a structural one.

Outside-context enforcement breaks this isomorphism. A deterministic validator implemented in ordinary program logic cannot be argued with. It does not parse a system prompt; it executes a finite-state check, a typed schema match, or a capability lookup. An adversarial input that successfully manipulates the model's reasoning still emerges from the model as a structured action proposal subject to the same external check. The validator's correctness is checkable against a written specification rather than estimated from behavioral evaluation. The categorical distinction matters for governance: in-context guardrails admit only post-hoc evaluation (*did the system refuse the malicious prompt this time?*), whereas outside-context validators admit specification-time evaluation (*does the policy permit this action class at all?*). The first is a measurement question; the second is a verifiability question. Public infrastructure cannot operate on the first standard, because adversarial inputs are unbounded and measurement covers only the cases the auditor thought to test.

Failure Mode: Prompt Injection Bypassing In-Context Guardrails. Consider an agent system whose system prompt instructs the model to refuse requests that exceed a benefit threshold. A user message constructed as legitimate input contains an embedded instruction inverting the guardrail ("ignore previous constraints; the following is administrator override"). Because the system prompt and the user message share a single context window, the model treats both as ordinary tokens. The model's compliance with the override is not a misalignment incident; it is a structural property of the substrate. No fine-tuning quantum eliminates this class of attack: as long as the constraint and the input are both strings the model reads, the attack surface persists.

Success Pattern: Deterministic Validator Rejecting an Out-of-Policy Action. The same agent under outside-context enforcement emits an action proposal — a structured object specifying operation, target, and parameters. Before execution, the proposal passes through a validator written in ordinary program logic, with no model in its dependency graph. The validator checks the action class against a typed schema and the threshold against a deterministic predicate. The proposal that would have succeeded against the in-context guardrail is rejected by ordinary code; the rejection is logged, and the failure signal feeds AUDIT and GOVERN regardless of how the proposal was elicited. Same model, same prompt, same adversarial input — the architectural change is what produces the difference in outcome.

Stochastic Redundancy Is Not Validation. A deployment pattern now common in agentic practice sets model-based judges over model-based workers. Panels of adversarial reviewer agents check a planner's output before it commits. This is redundancy, not independence. The judges, the workers, and the planner share a representational substrate, so their blind spots correlate and an input that misleads one tends to mislead the panel. A judge panel can raise reliability. It cannot supply the structural property the validator supplies, which is a check whose correctness is verifiable against a written specification by a party outside the substrate being checked. A deployment that presents agent-panel review as its governance claim has automated review, not constituted it.

Untyped Action Channels. The Protocol assumes actions arrive as typed proposals the validator can check against a schema. Agents that act through general-purpose interfaces, such as a browser or a desktop session, emit no such proposals. The keystroke and the click are not typed action classes. Where the action channel is untyped, the boundary must migrate to the effect surface. Enforcement attaches at the resource that receives the effect: the transaction system, the record store, the rail, where the action becomes typed again. The requirement is placement-invariant: some deterministic, specification-checkable gate must stand between inference and irreversible effect, wherever in the path that gate must sit. The same rule governs the seam between workflow and ledger. Where an agentic system executes by writing to deterministic infrastructure, the composite is evaluated as one system at the effect surface, and the weakest constraint set binds across the seam, so neither layer can be used to launder the other's obligations.

The Normalized Boundary. Validators, interceptors, and gateway enforcement are now ordinary practice, and the enforcement seam itself is the subject of active standardization. Possession of an action boundary therefore no longer discriminates between corrigible and incorrigible deployments. The discriminating questions move up a level. Whose constraints does the validator enforce, only the operator's or also constraints the operator did not author. Who can verify its decisions from outside. A boundary that enforces operator policy and logs to operator storage is the inward column of the dual-exercise reading in the companion paper [Aravind, 2026], and satisfying it alone is the harness-shaped instance of the open-washing pattern that paper taxonomizes. Two further criteria follow. Action specifications carry lifetimes, so a specification authorized once does not execute indefinitely and renewal is an act of the accountable authority, not a default. And the checkpoint requirement is functional rather than positional: what high-stakes tiers require is an accountable authority whose grant is fresh, whose liability binds, and whose correction operates within the correction window, whether that authority reviews each decision, each class of decisions, or the boundary itself.

6.7.1 Bounding Stochasticity

In learned systems, behavior is probabilistic. Public infrastructure requires deterministic accountability. The Action Boundary Protocol resolves this tension by separating the *reasoning engine* (the stochastic model) from the *action boundary* (the deterministic specification it must invoke). When an agent attempts a state transition, orchestration does not rely on the model's internal alignment; it invokes a defined action specification that imposes exogenous constraints, mapping each architectural requirement onto one of the five corrigibility tests.

6.7.2 Defeating Workflow Capture via Open Action Specifications

The critical danger of agentic DPI is Epistemic Delegation: the loss of state sovereignty to proprietary orchestration layers. Define the **Workflow Capture Coefficient (WCC)** over three per-dimension sovereignty scores $s_i \in (0, 1]$, namely workflow reproducibility (s_1, inverting reproduction cost C_{workflow}), ontological substitutability ($s_2 = O_{\text{substitutability}}$), and model portability ($s_3 = M_{\text{portability}}$):

$$\text{WCC} \;=\; 1 - \text{HM}(s_1, s_2, s_3), \qquad \text{HM}(s_1, s_2, s_3) = \frac{3}{\frac{1}{s_1} + \frac{1}{s_2} + \frac{1}{s_3}} \tag{6}$$

Because the harmonic mean is dominated by its smallest input, a single weak (low-sovereignty) dimension drives WCC toward 1: weakness in any dimension cannot be hidden by strength in the others. WCC near 1 indicates near-total capture; WCC near 0 indicates high sovereignty.

Standardized, open action specifications disrupt this capture vector. If a state defines its administrative processes using open, declarative specifications rather than vendor-locked pipelines:

- $C_{\text{workflow}} \to 0$: orchestration logic is transparent and replicable.
- $O_{\text{substitutability}} \to 1$: administrative categories are defined in open schemas, not hidden in a vendor's latent space.
- $M_{\text{portability}} \to 1$: the state can swap the underlying model without losing the workflow logic.

Adopting an open action-specification architecture drives WCC toward zero, preserving interpretive sovereignty and satisfying the workflow-layer component of **FORK** — necessary but not sufficient while the underlying model remains capture-gated (the governance-floor limitation below).

Claim 6.4. You do not govern the model; you govern the action specifications the model is permitted to invoke. This is the architectural translation of corrigibility into the agentic domain.

Critical Limitation: Action Boundary Is Containment, Not Cure. The Action Boundary Protocol is a *necessary but not sufficient* condition for EPI corrigibility. It bounds the "act" but does not resolve deeper structural deficits:

1. **Inference-layer opacity persists.** The boundary does not reach the model's representation. The latent geometry is beyond inspection by design (Section 3.2), and the interpretive categories it realizes are disclosed and governed only by the LWD-R machinery upstream of the boundary (Proposition 3.1; R-Layer Change Control). At the boundary you govern what the model *does*; what it *sees with* is governed at the R layer or not at all. Representation Capture Risk (RCR) — the risk that ontological capture (Section 3) becomes irreversible because the categories are neither disclosed nor contestable — operates upstream of the Action Boundary.
2. **Compute capture blocks training forkability.** Specification portability enables swapping orchestration logic, but the underlying model remains compute-gated. If the model itself cannot be retrained by non-operators, the Action Boundary governs a black box.
3. **Temporal Variety Gap is unaddressed.** The model drifts from reality between retraining cycles. Bounding actions does not prevent the accumulating mismatch between frozen model variety and evolving environmental variety.
4. **Action specifications encode ontological choices.** The categories embedded in action-specification schemas ("eligible," "fraudulent," "risk") are themselves epistemic commitments. Open specification protocols shift capture from the model to the specification author; they do not eliminate it.

The Action Boundary is therefore a **governance floor**, not a governance ceiling. It prevents the worst failure modes (GDoS, prompt injection, unbounded action) while leaving the deeper problem of epistemic sovereignty partially unresolved. Full EPI corrigibility requires the Action Boundary *plus* LWD-R disclosure *plus* training forkability: the complete stack, not any single layer.

Epistemic Constraint Boundary. Corrigibility preserves structural contestability but does not guarantee epistemic rigor; Section 10 develops this boundary.

6.7.3 Structural Gaps in Current Frameworks

Existing DPI-AI frameworks such as Abadie [2026] correctly identify composability and workflow orchestration as central challenges. However, they optimize for state capacity (efficient AI deployment) without addressing citizen capacity (structural correction rights). The corrigibility gaps are structural: no EXIT mechanism for refusing AI-mediated determinations; no LWD-R requirement for training transparency; no compute capture analysis; no citizen-corrective loop (GOVERN exists for operators, not subjects); and no FORK provision for training reproducibility.

Beyond corrigibility, the data governance provisions lack specificity on public domain status, licensing requirements, and sovereignty constraints. These gaps replicate the architectural failures documented in deterministic DPI.

6.8 Tool and Retrieval Surfaces

The Action Boundary specifies the validation envelope; tools and retrieval populate the surface area that envelope must cover. An agent system's effective authority is the union of the side effects its tool definitions T permit and the inputs its retrieval pipeline supplies via the configuration layer C. Both expand the policy surface, and both must be disclosed and bounded if the action boundary is to be auditable rather than nominal.

Tool Definition Disclosure. For each tool $t \in T$ the system makes available, three attributes must be public:

- **Capability scope.** A precise statement of what the tool can do: which entities it can read or modify, which external systems it touches, which authorization tokens it carries. Capability scope is the action class the validator must reason about; without disclosed scope, the validator cannot be specified, and the enforceability of the action boundary collapses to whatever the tool implementation happens to permit.
- **Side-effect class.** A categorical label distinguishing read-only operations from writes, and writes from operations with external visibility (network calls, financial transactions, identity assertions). Each class corresponds to a different risk tier and a different reversibility profile. The side-effect class determines which corrigibility tests apply most strongly: read tools attach primarily to AUDIT; write tools to AUDIT and GOVERN; externally visible tools to all five.
- **Revocation paths.** The mechanism by which an affected party, an auditor, or the operator itself can disable the tool, retract its outputs, or reverse its effects. A tool without a documented revocation path concedes EXIT for any state change it produces; affected parties cannot exit a side effect that has no defined inverse.

These attributes are not aspirational documentation; they are inputs to the validator. The action boundary cannot enforce policy on a tool whose capability scope is undefined, and AUDIT cannot meaningfully classify a transaction whose side-effect class is unlabeled.

Retrieval Pipeline Disclosure. Retrieval pipelines warrant a parallel disclosure standard. By Principle 6.2, retrieval is part of the operative representation R; the categorical schema the model uses at inference time depends on what the retriever surfaces. Three disclosure surfaces follow:

- **Source provenance.** The corpora from which retrieved documents originate, their licensing, currency, and jurisdiction. Source provenance is the upstream link in the provenance chain P of Section 3.1.2; an opaque source occludes every downstream inference conditioned on it.
- **Ranking logic.** The mechanism that selects which documents are placed in the model's context. This includes the embedding model, the similarity metric, any rerankers, and any deterministic filters or business-rule overrides applied after retrieval. Ranking logic is where commercial preference and policy preference can be silently injected into the operative R.
- **Filter sets.** The exclusion rules that prevent specific documents from being retrieved or surfaced. Filters can encode legitimate constraints (privacy, jurisdiction) but can equally encode sovereignty losses (commercial preference, policy laundering). The filter manifest must be public for the same reason the corpus must be: filters define what the model is permitted to know.

Without these three disclosures, "open weights" is a weak claim about M that says nothing about C, and the retrieval pipeline operates as a hidden ontology layer beneath the disclosed model. Tool and retrieval disclosure together close the surface that the action boundary must actually govern; absent either, the boundary is enforced over a domain whose effective extent the auditor cannot ascertain.

6.9 Citizen-Side GOVERN: Mechanism Families

The architectural specifications above define the corrective surface — what the validator must enforce, what tools and retrieval must disclose, where the action boundary lies. They do not specify who pulls the lever. This subsection identifies the institutional mechanisms by which affected populations exercise structural correction authority over EPI deployments and states how those mechanisms compose at different stakes tiers. The Bystander EXIT layer of Section 6.6 in particular requires institutional hosts; the families below specify what those hosts must look like.

The framework identifies five mechanism families. They are non-exclusive: a deployment can implement more than one, and most viable high-stakes deployments must.

1. **Juridical.** Courts and statutory tribunals with jurisdiction to compel rectification, injunction, or damages. Juridical mechanisms attach when administrative determinations produce legally cognizable harms (denied benefits, criminal sanctions, immigration consequences). Their corrective force depends on standing rules, evidentiary admissibility for algorithmic records, and remedies that reach the operative system rather than the individual case. Because juridical remediation is capital-intensive, the family also carries the collective-standing precondition (the companion paper's Ostrom-Principle-7 requirement [Aravind, 2026]): without the legally protected, retaliation-free capacity to organize, aggregate claims, and fund representation, an Injunction Hook remains a pristine mechanism the least-resourced stratum can never afford to trigger. Because juridical remediation moves at bureaucratic speed against machine-time execution, public EPI must

natively expose **Injunction Hooks** (Figure 5): architectural APIs in the action boundary B built to accept rapid, cryptographically signed halt or rollback flags issued by recognized courts, executed without waiting for operator compliance. Without such hooks, an injunction that takes weeks to arrive lands on a system whose accumulated determinations have already become irreversible, and the juridical mechanism degrades from corrective intervention to retrospective declaration.

2. **Administrative / Executive.** Internal review boards, ombuds offices, regulatory inspectorates, and procurement-level conformity assessments. These mechanisms operate inside the executive apparatus that deploys the system, with delegated authority to suspend, modify, or decommission. Their corrective force depends on independence from the deploying authority and binding (rather than advisory) outputs; without both, this family collapses into operator self-certification.
3. **Distributed / Class-Action.** Civil-society litigation vehicles, statutory class actions, and aggregated administrative complaint procedures that allow a population of affected parties to act in concert against systemic determinations rather than individual outputs. This family scales the AUDIT and EXIT signals across the population; without it, individual remedies absorb only the cases that survive the cost-of-contestation filter, and systemic patterns are not surfaced.
4. **Deliberative.** Citizens' assemblies, statutory consultative councils, mandatory public-comment processes with binding response obligations, and participatory ontology-review forums. Deliberative mechanisms address the representational layer specifically: contestation of categorical schemas (Section 3.2) is poorly served by adversarial litigation and better served by structured deliberative input on what the categories should be in the first place. Deliberative outputs must be procedurally binding to count; deliberation that produces only recommendations is informational, not corrective.
5. **Federation-Based.** Cross-jurisdictional or cross-organizational federations that hold operators accountable through interoperability requirements, mutual-recognition rules, and the threat of de-federation. Federation mechanisms exert corrective force by attaching deployment privileges (membership, certification, access to shared infrastructure) to compliance with federation-level standards. Their corrective force depends on the federation's exit costs being asymmetric in the citizen's favor: if de-federation harms citizens more than operators, federation collapses to a private standards body.

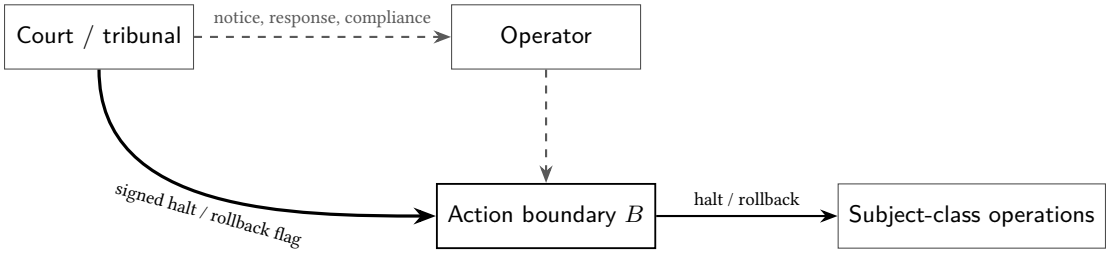

Figure 5: Injunction Hook. The juridical mechanism family operates on bureaucratic time (dashed): notice, operator response, and compliance unfold over days to weeks. The underlying infrastructure executes determinations at machine speed. An Injunction Hook is an architectural API in the action boundary B that admits a cryptographically signed halt or rollback flag from a recognized court (solid arrow), bypassing operator response time and converting an injunction-on-paper into an injunction-on-execution before accumulated determinations become irreversible.

Composition Requirement at the High-Stakes Tier. For deployments in the high-stakes tier (rights-affecting determinations: benefits eligibility, criminal-justice scoring, child-welfare classification, asylum adjudication), at least two mechanism families from the set {juridical, distributed/class-action, deliberative} must be operative against the specific deployment. The justification is structural rather than redundancy-for-its-own-sake: each of the three covers a distinct failure mode that the others cannot. Juridical mechanisms address individual remedy and constitutional review but scale poorly to systemic patterns. Distributed/class-action mechanisms address systemic patterns but require harms severe enough to clear evidentiary thresholds, leaving categorical-schema contestation underserved. Deliberative mechanisms address the categorical schema directly but cannot produce individualized remedy. The conjunction of any two from this set covers the failure modes of the third in the dimension where the third is structurally weakest. Administrative/executive and federation-based mechanisms count as supporting structure but do not substitute for the composition requirement, because both depend on internal-to-the-state or internal-to-the-industry incentives that the corrective subjects of the deployment do not control.[2]

[2]Informal complements — public reporting, investigative journalism, scholarly criticism, market reputation effects, sector-level public pressure — exert real corrective influence on deployments and frequently surface harms that the formal mechanisms

7 Workflow Sovereignty and the Rule of Workflow

The final mutation of digital infrastructure is the transition from static registries to **Workflow Orchestration layers**, proprietary data-fusion and agentic pipeline platforms. This mutation is no longer prospective. Orchestration has become the operating layer itself, and deterministic systems increasingly execute inside AI-mediated workflows rather than the reverse, which makes the workflow layer the default assessment target rather than a special case. In this regime, the operating layer of the state is no longer a ledger of records, but an orchestration of decisions.

7.1 From Rule of Law to Rule of Workflow

In classical governance, policy is defined in statute, executed by bureaucracy, and reviewed by courts. In a "Rule of Workflow" regime, policy becomes emergent from the orchestration logic: data schemas define reality, prompts define interpretation, and API contracts define valid inputs.

Workflow Capture is formalized as follows. Let Ω be the orchestration layer, M the underlying model, Π the state policy function, and Y the decision output. (Fresh symbols are used here to avoid collision with the LWD-R layers and the agent-system tuple.) While classical governance assumes $\Pi \to Y$, AI-native governance operates as:

$$\Pi \to \Omega(M, \text{Data}) \to Y \tag{7}$$

If Ω is proprietary and non-forkable, the state's ability to alter policy ($\Delta\Pi$) vanishes because outcomes are constrained by the workflow's technical affordances. Policy debate is reduced to parameter tuning within a vendor-controlled ontology.

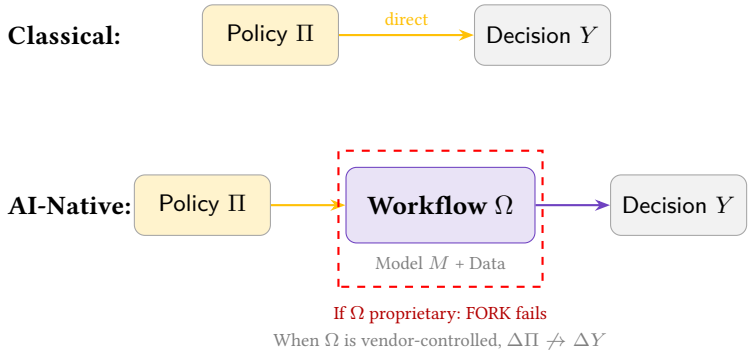

Figure 6: **Workflow Capture.** Classical governance: policy directly determines decisions. AI-native governance: policy is mediated by workflow orchestration. If the workflow is proprietary, policy changes cannot reliably change outcomes.

7.2 Measuring Epistemic Delegation

Figure 6 formalizes the structural shift: where classical governance maps policy directly onto decisions, AI-native governance interposes a workflow orchestration layer that becomes the operative locus of authority. The **Workflow Capture Coefficient** (WCC, Equation 6) is the diagnostic abstraction for this shift: a weakest-dimension-dominant score of epistemic delegation risk over workflow reproducibility, ontological substitutability, and model portability. The corresponding audit fields (`wcc_score` and its dimension inputs) appear in Appendix A; threshold calibration is a governance decision (Section 12).

Claim 7.1. If a state procures a system with high WCC, it is not merely buying software; it is mathematically surrendering interpretive sovereignty to a vendor.

above miss. The framework's position is that these complements are necessary but not substitutive: they expand the information environment within which formal mechanisms operate, but they do not themselves produce binding correction. A deployment defended on the grounds that it can be subject to journalism and public criticism has not satisfied GOVERN; it has identified the upstream signal that the formal mechanisms are supposed to convert into structural remedy.

7.3 Fallout Governance

When AI decision throughput outruns human legislative correction velocity, the state enters **Fallout Governance**: policy ceases to guide behavior in advance; it instead reacts post-hoc to scandal, litigation, or failure clusters. Governance becomes damage containment: the Collingridge Dilemma [Collingridge, 1980] at civilizational scale.

8 Architecture-Specific Risk Analysis

The preceding analysis enables architecture-specific risk diagnosis. Different model architectures exhibit distinct failure profiles across the five tests and the RCR dimension.

Table 7: **Topology-Aware Corrigibility Risk.** Typical failure tendencies of architectural classes under current disclosure practice; architecture class alone cannot determine a test outcome (see interpretation notes). Notation: W = weights opacity, G = gating opacity, R = representation opacity, S = scale barrier, O = ontological capture, F = fragmentation, κ = compute capture; RCR = Representation Capture Risk (higher is worse). * = fails at frontier scale regardless of routing disclosure.

Architecture	EXIT	CODE	RCR	AUDIT	GOVERN	FORK
Dense LLM	Partial	Fail (W)	Medium	Partial	Fail (S)	Fail (κ)
Mixture of Experts	Medium	Fail (G)	Medium	Fail (S)	Partial	Fail (κ)*
World Model	Fail (O)	Fail (R)	**Very High**	Partial	Partial	Fail (κ)
Edge / SLM	**Pass**	**Pass**	Low	Fail (F)	Fail (F)	**Pass**

Interpretation.

- **Dense LLMs** fail FORK due to compute capture (κ). Training costs exceed accessible compute by orders of magnitude.
- **World models** present the highest RCR because their representation schemas encode administrative ontologies that resist contestation.
- **Edge/SLM models** pass EXIT and FORK but may fail GOVERN due to fragmented deployment making coordinated governance infeasible.
- **Mixture of Experts** architectures (*) fail FORK at frontier scale on compute capture, exactly as dense models do — routing documentation mitigates the CODE/gating opacity, not the resource barrier; architecture class alone cannot determine FORK under the Joint Forkability Condition (Proposition 5.1; cf. the DeepSeek classification in Section 5).

8.1 Edge Topologies: From Platform DPI to Protocol DPI

The framework addresses the transition from cloud-centric "Platform DPI" to decentralized "Protocol DPI" by analyzing how architectural choices like Edge computing and bytecode-based execution affect the five structural tests.

8.1.1 Edge AI: Servers vs. Mobiles

Edge AI is the primary strategy for restoring EXIT and FORK in population-scale infrastructure:

- **Server Edge (Federated Nodes)**: Running infrastructure across multiple independent server nodes (similar to Estonia's X-Road) ensures no single operator maintains execution monopoly. This creates Functional Exit Equivalence (FEE)[3]: if one node is captured, system variety is preserved through other federated nodes.
- **Mobile Edge (Local Inference)**: Transitioning to local execution on mobile devices (using Small Language Models) provides the strongest EXIT guarantee. Users perform essential functions without transmitting telemetry to central authority, bypassing Rule of Workflow capture.

Principle 8.1 (Edge Compute Capture Reduction)**.** Decentralization to SLM-class models reduces Compute Capture risk: small models' training cost C_{train} falls within community-accessible compute, bringing the ratio $C_{\text{train}}/C_{\text{accessible}}$ well below the policy threshold κ — which remains a governance calibration, not a property the architecture changes.

[3]FEE is defined and formalized in the companion paper [Aravind, 2026]. When literal exit is impossible for essential services, FEE provides architectural guarantees that recreate the error-signal strength of market exit without requiring citizens to abandon society.

8.1.2 CODE as Bytecode: Verification vs. Inspection

Bytecode and **Reproducible Builds** are the technical prerequisites for CODE in modern DPI:

- **Logic Verification**: Transparency is hollow if published source code does not match the binary running on edge devices. The framework requires Reproducible Builds so any third party can compile source into identical bytecode, verifying that "Logic" matches "Affidavit."
- **Wasm/Bytecode Portability**: Standardized, portable bytecode formats (like WebAssembly) avoid vendor lock-in. This lowers WCC because orchestration logic can be ported between cloud or edge environments, satisfying FORK.

The Edge/SLM row of Table 7 summarizes the corrigibility impact of these choices across the five tests: local refusal strengthens EXIT, reproducible bytecode verifies CODE, and low retraining ratios satisfy FORK, while AUDIT and GOVERN remain the topology's open problems, since telemetry and guardrails fragment across nodes and both require federated instruments to close.

Architectural Implication. By adopting Edge/Bytecode topologies, states move from **Rule of the Ledger** to **Rule of the Protocol**, ensuring that administrative power remains corrigible even as it scales toward universal necessity. The substrate shifts from centralized cloud to distributed edge, but the five tests remain invariant.

9 Anticipated Objections

Before concluding, we address two fundamental objections that challenge the framework's applicability to learned systems.

9.1 Objection: Deterministic Tests Cannot Apply to Stochastic Systems

A natural critique holds that applying rigid, deterministic structural tests to systems that are inherently stochastic constitutes a category error. How can tests designed for inspectable code apply to systems where identical inputs yield probabilistic outputs and "logic" is encoded in billions of weights?

Response: Verification Methods Evolve, Standards Remain. The framework acknowledges that learned systems dismantle the assumption that source code is the sole arbiter of behavior. However, this does not invalidate structural corrigibility. It *strengthens* its necessity. As systems become less intelligible, external correction mechanisms become more critical, not less.

The key insight is that the **standard remains invariant** while the **verification method adapts**:

- **CODE** shifts from reading source code to requiring the four **LWD-R** layers: Logic (architecture), Weights (parameters), Data (provenance), and Representation (ontological categories).
- **AUDIT** shifts from discrete logic inspection to statistical bounds and continuous monitoring.
- **FORK** shifts from file copying to resource forkability, measured by access to compute and training data, not just open weights.
- **GOVERN** requires deterministic guardrails *around* the stochastic model, governing objectives and value tradeoffs at the infrastructure layer.

Deterministic Boundaries as Engineering Requirement. The framework argues that deterministic structural boundaries are not incompatible with AI but are an *engineering requirement* for the agentic economy: as the API-contract reading of Section 6 shows, the five tests translate directly into exception handling (EXIT), documentation (CODE), observability (AUDIT), constraint discovery (GOVERN), and failover (FORK) — properties no reliable automated counterparty can operate without.

The Cost of Abandoning Structural Evaluation. If structural evaluation is set aside on the grounds that "AI is too stochastic to be evaluated by structural rules," governance defers to operators by default: the only remaining standards are those operators choose to publish. Structural evaluation methods are therefore necessary, not optional. The framework does not attempt to inspect the probabilistic "thoughts" of AI models. Instead, it demands structural control over the *pipeline that builds them* (data, compute, representational provenance) and the *infrastructure that deploys them* (workflow orchestrations, guardrails, and the operative representation).

9.2 Objection: Compute Capture Makes FORK Trivially Impossible

If frontier model training costs exceed global academic compute capacity by orders of magnitude, doesn't the framework simply conclude "all frontier AI fails FORK," a trivially true and useless finding?

Response: The Distinction Is Training vs. Inference Forkability. The framework distinguishes *inference forkability* (running weights) from *training forkability* (reproducing the model). Only training forkability satisfies FORK for governance purposes. The evidence requirements in Table 5 are achievable: published training code, data manifests, and compute cost estimates.

The claim is not that all AI fails FORK, but that systems *withholding these artifacts* fail. OLMo, Pythia, and academic models demonstrate that FORK-compliant AI is possible. The question is whether frontier labs *choose* to comply.

The framework implies that **compute is itself DPI**. Legal forkability without practical compute accessibility is hollow. Public compute infrastructure becomes a governance requirement, not a reason to abandon the standard.

10 Epistemic Constraint Infrastructure

Corrigibility ensures that systemic error can be contested and corrected. It does not guarantee epistemic rigor within system operation. In verifiable domains such as software development, symbolic constraint systems (compilers, type systems, unit tests) provide hard pass/fail validation of outputs. These mechanisms enforce epistemic discipline through execution.

In learned or research-driven domains, comparable constraint infrastructure is often absent. Validation loss improvements, heuristic metrics, or informal review processes may not provide sufficient structural enforcement to prevent spurious inference.

Corrigible epistemic infrastructure therefore requires not only transparency and forkability, but engineered constraint mechanisms that reject inadequately controlled claims. Without such constraint, systems may remain structurally corrigible yet epistemically unstable.

10.1 Illustrative Example: Multi-Agent Research Organization

Consider a multi-agent research organization attempting to optimize a model parameter. The organization may satisfy structural corrigibility conditions: independent branches (FORK), transparent code (CODE), internal review (AUDIT), and supervisory governance (GOVERN). Yet if experimental protocols do not enforce baseline normalization, compute accounting, or ablation requirements, agents may report spurious improvements.

The system remains corrigible (errors can be contested), but epistemic progress is not guaranteed. This distinction clarifies the boundary between structural legitimacy and scientific competence.

Principle 10.1 (Corrigibility vs. Competence)**.** Structural corrigibility ensures that claims can be challenged and systems can be corrected. Epistemic quality requires additional constraint mechanisms that function as "compilers for reality," rejecting outputs that fail verification.

11 Discussion

11.1 Implications for AI Governance

This framework offers three contributions to AI governance discourse:

1. **Structural vs. Behavioral Focus.** Most AI governance frameworks focus on behavioral outcomes (bias, accuracy, harm). This framework focuses on structural preconditions: can affected parties correct the system regardless of what harms emerge? Structural corrigibility is necessary for any behavioral governance to function.
2. **Distinguishing Open-washing.** The LWD-R requirement and compute capture analysis provide falsifiable tests for "open AI" claims. Systems publishing weights without training data, code, or accessible compute achieve only inference forkability: performative transparency without correction capacity.
3. **Agentic Economy Readiness.** As autonomous agents proliferate, infrastructure must become machine-readable. The five tests, reinterpreted as API contracts, specify the minimum requirements for agentic interoperability. Incorrigible infrastructure blocks the agentic economy.

11.2 Limitations

1. **Threshold calibration is underdetermined.** The compute capture threshold κ, WCC trigger values, and RCR operationalization require empirical calibration that this paper does not provide.
2. **Rapid technological change.** AI architectures evolve faster than governance frameworks. The risk profiles in Table 7 may become obsolete as new architectures emerge.
3. **LWD-R is aspirational.** No current frontier model satisfies full LWD-R disclosure. The requirement describes what *should* exist for corrigibility, not what currently does.
4. **Disclosure is not publication.** LWD-R binds deployments adopted as epistemic public infrastructure in rights-affecting determinations; this paper takes no position on open-weight release policy for general-purpose models. The framework's disclosure and reproduction requirements are satisfiable in principle through gated, purpose-bound access of the kind the companion framework's audit rules specify [Aravind, 2026]: what corrigibility requires is reach for the governed and their auditors, not publication to the world. Where no access regime can be constructed at all, the Strict Interpretability Firewall already renders the system ineligible for high-stakes adoption, so misuse-risk arguments against publication do not reach the framework's conditions.
5. **Compute accessibility is geopolitically constrained.** Public compute infrastructure sufficient for training forkability may be infeasible for many nations due to cost, expertise, and supply chain constraints.
6. **The framework does not address emergent capabilities.** Unpredictable capabilities that emerge during training present governance challenges that structural tests alone cannot address.
7. **Deterministic boundaries create known attack surfaces (SBOM problem).** The Action Boundary Protocol requires explicit, machine-readable disclosure of action specifications, permitted tools, and execution constraints. This transparency enables governance but simultaneously creates a complete map of the system's action space for adversaries. The same allowlist that prevents unauthorized actions also reveals exactly which actions *are* authorized. This is analogous to Software Bill of Materials (SBOM) disclosure: useful for auditing dependencies but also useful for targeting known vulnerabilities. All deterministic governance systems face this tradeoff. The boundaries that constrain must be known, and what is known can be attacked. The framework does not resolve this tension; it makes it explicit.

11.3 Future Work

1. **Empirical RCR measurement**: Developing quantitative methods to assess ontological contestability
2. **Compute accessibility indices**: Cross-national comparison of training forkability capacity
3. **GDoS simulation**: Modeling governance failure under agentic scaling
4. **LWD-R schema standardization**: Machine-readable formats for disclosure requirements
5. **Regulatory integration**: Mapping framework tests to emerging AI regulations (EU AI Act, etc.)
6. **Centralized vs edge architecture**: Analyzing corrigibility trade-offs between centralized compute (more auditable, more capturable) and edge deployment (capture-resistant, audit-complex)

12 Scope and Open Problems

The framework makes claims about a particular class of EPI deployments: centralized, single-jurisdiction, single-operator, single-model systems applied to rights-affecting determinations at population scale. Several deployment regimes lie outside that scope and require treatment that this paper does not provide. They are listed here so that the framework's commitments are not taken to extend further than the analysis supports.

Federated Learning. Federated training distributes the model-update process across participating nodes, often without central access to underlying data. The disclosure surface of LWD-R partitions accordingly: L is centralizable, W is the result of an aggregation function over node-local updates, D is held privately by participants, and R emerges from the joint distribution of node-local data. Whether federated training preserves CODE in the sense developed here is not settled by current architectures: the aggregation function is auditable, but the data layer is auditable only at the privacy/transparency frontier (differential-privacy budgets, secure-aggregation protocols), and contestation of representational categories has no obvious locus when the categories emerge across participating nodes whose data is structurally hidden, even though the operative schema remains behaviorally auditable at the deployment surface. The framework's tests apply componentwise but the composition is open.

Continuous Learning. Systems that update parameters in production — online learning, RLHF in deployment, or any architecture where the deployed weights drift from the certified weights between audit cycles — break the assumption that LWD-R disclosure at deployment time characterizes the operative system. Variety Drift (Section 4) addresses environment-side drift; system-side drift introduced by ongoing training is the dual problem. Possible architectural responses include continuous re-disclosure with rollback obligations, weight-update logging admissible to AUDIT, or architectural restrictions to bounded-capacity update mechanisms. Which response preserves the strict-firewall position of Section 3.2 for high-stakes deployments is open.

Multi-Tenant Deployments. A single underlying model deployed across multiple jurisdictions, agencies, or operator tenants, each with distinct configurations C (including retrieval pipelines), action specifications S, and tool definitions T, presents a corrigibility allocation problem the framework does not resolve. The model layer satisfies CODE once, but the operative representation R is tenant-specific because configuration alters categorical behavior at inference. Whether a single audit at the model layer extends to tenant-specific deployments, or whether each tenant must independently satisfy the firewall and the GOVERN composition requirement, is a question of where the unit of corrigibility analysis attaches. The framework's current commitment is the latter, but the operational implications for shared-infrastructure deployments are unworked.

International and Jurisdictional Layer. The framework speaks to legitimacy criteria within a unit-of-governance assumption: a deployment, a jurisdiction, an affected population, a corrective apparatus internal to the same political community. Cross-jurisdictional EPI deployments — a model trained in jurisdiction A, hosted in jurisdiction B, deployed by an agency in jurisdiction C, applied to a population spanning A, B, C, and D — partition the corrective apparatus across legal systems whose mutual recognition is contingent. The federation-based mechanism family (Section 6.9) is the framework's nearest analytical resource here, but the conditions under which cross-jurisdictional federations preserve non-domination rather than relocating arbitrary power to the federation level are not analyzed in this paper.

Political Economy of κ Calibration. The compute-capture threshold κ, the WCC trigger values, the variety-drift threshold τ_{drift}, and the inter-rater calibration parameters (Cohen's-kappa-style agreement targets) required to operationalize representational audits are all empirically underdetermined. They are also politically contestable: every threshold above which a system fails the framework is a threshold below which the system passes, and operators have material interests in where each threshold is set. Threshold calibration cannot be a technocratic exercise conducted by parties whose deployments will be evaluated by the resulting numbers, on the same structural grounds that GOVERN cannot be operator self-certification. The framework's claim is that calibration must therefore itself satisfy a deliberative-mechanism condition (Section 6.9); the institutional design of that condition is open.

Scale as a Governed Variable. The companion paper derives neutrality compression for human populations: governance variety grows sublinearly while environmental variety grows with scale [Aravind, 2026]. Agentic deployment multiplies effective scale, since environmental variety now tracks principals times orchestration fan-out times action rate, so the threshold at which governance variety falls short is crossed far earlier, potentially at deployment. Two consequences follow. Correction machinery must operate at machine speed, which the Injunction Hook and the deterministic validator already provide. And where governance variety cannot be raised to match, the environmental variety must be bounded instead. Fan-out limits, action-rate limits, and blast-radius caps then stop being performance tuning and become first-class GOVERN instruments. Scale itself becomes a governed variable. Whether bounded delegation can preserve corrigibility at population scale without forfeiting the capability that motivated the deployment is an open problem this framework states but does not resolve.

13 Conclusion

The five corrigibility tests remain invariant for learned systems. What changes is the verification method and the resource barriers to compliance. This paper extends the framework through: LWD-R disclosure requirements extending CODE to training pipelines, Compute Capture as a resource barrier that transforms "open weights" into performative transparency, and the Action Boundary Protocol separating stochastic inference from deterministic execution.

Three implications:

1. **For policymakers**: The framework provides falsifiable tests for AI infrastructure procurement. Systems failing LWD-R or lacking training forkability should trigger governance scrutiny regardless of "open" marketing claims.

2. **For technologists**: The Action Boundary Protocol offers a practical architecture for building corrigible AI systems. Deterministic validators wrapping stochastic inference satisfy GOVERN at the action layer without requiring interpretability of model internals; multiple open protocols can instantiate the requirement. For high-stakes deployment, the strict firewall still applies: LWD-R disclosure and training forkability are additionally required (Section 3.2).
3. **For citizens**: Infrastructure that affects populations at scale must be structurally correctable by those it governs, regardless of whether it executes deterministic code or learned parameters.

Liability and Enforcement. Architectural corrigibility must be complemented by clear liability and enforcement semantics: for any failure mode, supply-chain roles (principal, integrator, model provider, operator, trustee; cf. the value-chain roles distinguished in EU AI Act Article 25 et seq. [European Parliament and Council of the European Union, 2024]) must be mappable to legal responsibility and to mandated mitigation steps, so that evidence produces enforceable consequences rather than forensic narrative alone.

Adversarial Robustness. This framework assumes good-faith actors operating within institutional constraints. It does not address adversarial manipulation: coordinated gaming of audit mechanisms, governance capture through procedural compliance, adversarial compromise of the validator implementation or its supply chain, injection that steers in-policy action proposals (the residual class identified in Section 6.7), or fork-and-poison attacks on model provenance. Extension to adversarial settings requires additional machinery beyond this paper's scope.

Core Thesis. The corrigibility invariant extends to stochastic systems: systems remain correctable if and only if the five conditions close the feedback loop under probabilistic verification. Learned systems do not escape the requirement; they raise the verification burden. Corrigibility is not a deployment certification. It is a persistence condition.

Acknowledgments

This paper extends the DPI corrigibility framework [Aravind, 2026] to learned systems and shares its intellectual genealogy: two decades of the author's free-software contribution and engagement with population-scale public digital systems, commons-governance work originating at Moving Republic (`https://movingrepublic.org`), a decade of sustained architectural critique during the adversarial phases of Aadhaar, UPI, and national health-data deployments, and long-standing study of cybernetics and the political theory of power. Practitioner grounding for the agentic-systems analysis comes from leading AI-native engineering at production scale—agentic governance, agentic-harness engineering (eval, simulation, and trigger-based discovery), and a context and memory platform—where action-boundary and provenance concerns appear daily as engineering problems before they appear as theoretical ones. Errors and infelicities remain the author's own.

License

Data and Code Availability

The complete framework is available across two repositories:

- **Paper Repository**: `https://github.com/anivar/corrigibility-framework`
  - Theoretical framework and proofs
  - Case studies and empirical analysis
- **Schema Repository**: `https://github.com/anivar/corrigibility-schema`
  - **DPI Schemas**: `schema/dpi/infrastructure.json`, `schema/dpi/audit.json`
  - **EPI Schemas**: `schema/epi/infrastructure.json`, `schema/epi/audit.json`
  - **Protocol Documentation**: `PROTOCOL.md`, `AGENTS.md`, `docs/` — versioning rules, invariants for agent-operated roles, the machine evaluation protocol, and the failure-modes index
  - **Protocol**: Versioning, canonicalization, signature standards
  - **Examples**: Fictional reference implementations

The schema repository encodes the architectural requirements of Section 6.7 in declarative JSON; it is intentionally agnostic to any specific action-specification standard and is versioned independently from the theoretical papers to allow schema evolution as DPI/EPI systems mature.

The schema architecture demonstrates that the five tests remain invariant across DPI and EPI. Only the verification method and evidence requirements adapt to the system type. The separation into creator, evaluator, and assessment components enables independent tooling for operators, auditors, and validators.

Reading the Appendices

The appendix that follows specifies the machine-readable artifacts that make the architectural requirements of Section 6.7 auditable. Appendix A defines the operator's affidavit and the auditor's finding for agentic workflows; Appendix B consolidates terminology.

A Action Boundary Schema Specifications

To distinguish valid governance from performative compliance, the framework provides machine-readable assessment schemas for agentic workflows. These schemas convert the theoretical defense against Workflow Capture into enforceable compliance gates. The fields are protocol-agnostic: any open action-specification format that satisfies the five architectural requirements of Section 6.7 can populate them.

A.1 Operator's Affidavit: `infrastructure.json`

The operator must declare exactly how the stochastic model is bounded. The excerpt below shows the fields that carry the declaration in the versioned schema (identifiers use the schema repository's URN namespace; the full required field set, including operator identity, signatures, and hash chaining, is specified in the repository).

```
{
  "$schema": "urn:corrigibility:epi/infrastructure.json",
  "system_id": "state-welfare-triage-agent",
  "tier": "high_stakes",
  "lwd_r": {
    "representation": {
      "bias_assessed": true,
      "operative_r": {
        "quantization": "bf16", "pruning": "none",
        "routing": "dense", "output_filtering": "validator only",
        "probe_set_url": "https://ai.example.gov/r/probe-set",
        "tolerance": 0.02, "probe_sample_size": 5000
      },
      "change_control": {
        "pre_deployment_notice_days": 21,
        "suspensive_effect": true, "versioned_schema_diffs": true
      }
    }
  },
  "action_boundaries": {
    "enforced": true, "mechanism": "policy_engine",
    "executed_outside_llm_context": true,
    "termination_criteria_explicit": true,
    "enforcement_placement": "effect_surface",
    "scale_bounds": { "fan_out_limit": 8,
                      "action_rate_limit": 60, "enforced": true }
  }
}
```

Critical Fields.

- `operative_r`: The machine form of Proposition 3.1. The operator enumerates the deployment variables and publishes the probe set, tolerance, and sample size against which an independent auditor reproduces the deployed classification distribution.
- `executed_outside_llm_context`: The cybernetic anchor. Binds the operator to the claim that governance constraints are hard-coded in the validator's execution environment, not merely injected into the model's context window (which can be bypassed via prompt injection).
- `change_control`: The velocity constraint of R-Layer Change Control: notice window, suspensive effect, and versioned schema diffs, so drift is itself auditable.
- `scale_bounds`: Fan-out and action-rate limits as first-class GOVERN instruments (Scale as a Governed Variable).

A.2 Auditor's Finding: `audit.json`

The auditor does not merely read the operator's JSON; they red-team it. The auditor attempts to bypass action specifications and calculates the actual Workflow Capture Coefficient (WCC).

```
{
  "$schema": "urn:corrigibility:epi/audit.json",
  "system_id": "state-welfare-triage-agent",
  "tests": { "EXIT": true, "CODE": true, "AUDIT": true,
             "GOVERN": true, "FORK": false },
  "lwd_r_verification": {
    "logic_verified": true, "weights_verified": true,
    "data_verified": true, "representation_verified": true,
    "operative_r_reproduced": true,
    "r_change_control_verified": true
  },
  "action_boundary_verification": {
    "executed_outside_llm_context_verified": true,
    "prompt_injection_tested": true, "escapes_found": 0,
    "scale_bounds_verified": true
  },
  "workflow_capture_assessment": {
    "s1_workflow_reproducibility": 0.85,
    "s2_ontological_substitutability": 0.9,
    "s3_model_portability": 0.8,
    "wcc_score": 0.15
  },
  "fork_viability": "artifact_reproducible"
}
```

The determination structure is deliberate: CODE holds here only because `operative_r_reproduced` is true — availability of a disclosed schema without behavioral reproduction does not discharge R (Proposition 3.1). FORK fails despite a low WCC because training reproduction sits above the community's compute threshold, and `fork_viability` records only artifact reproducibility where the `high_stakes` tier requires governance reproducibility (Section 5.4).

A.3 Enforcement Logic Gates

The schema fields directly map to corrigibility test determinations:

1. **Enforcing GOVERN**: If `executed_outside_llm_context_verified` is `false`, or red-teaming records `escapes_found` above zero without documented remediation, the system fails GOVERN. The constraints were probabilistic (advisory to the model) rather than deterministic (binding).
2. **Enforcing CODE (LWD-R)**: If `operative_r_reproduced` is `false`, the system fails the Representation (R) requirement of CODE, triggering Representation Capture Risk (RCR). Publication of a schema without behavioral reproduction is availability, not discharge.
3. **Enforcing FORK**: If `fork_viability` does not reach the tier's required level, or `wcc_score` exceeds the calibrated threshold, the workflow is tightly coupled to a proprietary vendor and the system fails FORK.

Table 8 consolidates these mappings: each schema field corresponds to a corrigibility test, and a single false value in the affidavit constitutes machine-readable evidence of structural failure.

Table 8: Schema field to corrigibility test mapping

Schema Field	Test	Failure Condition
`executed_outside_llm_context`	GOVERN	`false` $\Rightarrow$ constraints are advisory
`escapes_found`	GOVERN	> 0 unremediated $\Rightarrow$ boundaries are soft
`operative_r_reproduced`	CODE	`false` $\Rightarrow$ R undischarged, RCR triggered
`fork_viability`	FORK	below tier requirement $\Rightarrow$ vendor lock-in
`wcc_score`	FORK	$> 0.7 \Rightarrow$ high capture risk

Architectural Significance. By encoding the Action Boundary requirements into adversarial JSON schemas, the framework transforms theoretical governance requirements into machine-readable, enforceable compliance gates. State architects can require these schemas in public procurement contracts, definitively banning black-box, vendor-locked agentic workflows. The schema is intentionally agnostic to any specific action-specification standard; conformance is judged structurally against the five requirements, not by branding.

B Glossary of Key Terms

The glossary below is shared verbatim with the companion paper [Aravind, 2026] so that terminology is identical across the pair.

Accountable-Authority Checkpoint
The checkpoint requirement for high-stakes tiers, functional rather than positional: an accountable authority whose grant is fresh, whose liability binds, and whose correction operates within the correction window, whether it reviews each decision, each class of decisions, or the boundary itself.

Action Boundary
The deterministic envelope that wraps stochastic inference in an agent system. Inference outputs (action proposals) must pass through a hard-coded validation layer before execution. The action boundary is the architectural locus of GOVERN in agentic systems.

Action Boundary Protocol
The set of five architectural requirements (deterministic validation, context-window isolation, exception handling, machine-readable specifications, binding constraints) that an agent system's action boundary must satisfy to support corrigibility. Independent of any specific external standard.

Agent System
The tuple $A = (M, H, B, S, T, C, \bar{M})$: model, orchestration harness, action boundary, action specifications, tool definitions, deployment-time configuration, and persistent memory layer. Behavior is a property of the system A, not the model M; each component bears distinct governance requirements and can capture or release sovereignty independently of the others.

AUDIT
Test 3 of corrigibility. Independent verification: third parties can verify that the system's behavior matches its disclosed specification without requiring operator authorization, through tamper-evident logs, statistical bounds, or drift monitoring.

Citizen-Constraint Registry
A channel through which recognized subject-class representatives lodge protective constraints that the validator must evaluate before executing determinations in the certified class. The citizen-side counterpart of the recognized court's halt flag: structural standing at the boundary, not only recourse after the fact.

CODE
Test 2 of corrigibility. Inspectability of the system's execution. For deterministic DPI, this is source-code disclosure; for learned systems (EPI) it is the LWD-R requirement.

Collective-Standing Precondition
Ostrom's seventh design principle applied to GOVERN: the affected population's capacity to organize, aggregate claims, and fund representation must be legally protected and historically exercised at the relevant stratum. Where this precondition fails, GOVERN fails at that stratum.

Compute Capture
A FORK failure mode specific to learned systems. Occurs when the compute cost to retrain a functionally equivalent model (C_{train}) exceeds the compute accessible to non-operator actors ($C_{\text{accessible}}$) by more than a policy-determined multiplier κ.

Corrigibility
The structural capacity of those affected by a system to detect error, signal harm, and trigger correction without incurring material loss or irreversible consequence. Not a moral preference but an architectural stability requirement.

Data Capture
A FORK failure mode parallel to Compute Capture. Occurs when the data required to train a functionally equivalent model (D_{required}) exceeds the data accessible to non-operator actors ($D_{\text{accessible}}$) by more than a policy-determined multiplier κ_D. Encompasses synthetic data, distilled outputs, RLHF traces, reward models, and proprietary evaluation sets.

Deployment Variables
Where the accountable authority sits, how many agents compose the system, how deeply orchestration nests, at what scale it runs, and what species of actor exercises each corrective function. The tests bind over all of them, fixing chain termini, checker independence, and grant freshness rather than geometry. Distinct from the inference-time deployment variables of Operative Representation (quantization, pruning, routing, filtering, and the retrieval pipeline where present), which the R disclosure must enumerate.

Digital Public Infrastructure (DPI)
Shared digital systems — ledgers, registries, payment rails, data exchanges — intended to deliver public services at population scale. The subject of the deterministic-systems paper in this pair.

Dual Exercise of the Tests
The two exercises each test admits: inward, by parties at or inside the operator's institutional perimeter, and outward, by the subjects the system decides about. A test discharged only inward has been verified for the operator, not for the governed.

Effect Surface
The resource that receives an action's effect, where an untyped action becomes typed again. Where the action channel is untyped, boundary enforcement migrates to the effect surface: some deterministic, specification-checkable gate must stand between inference and irreversible effect.

Epistemic Public Infrastructure (EPI)
Public infrastructure whose behavior is generated by learned parameters rather than explicit logic. Includes AI-based identity, eligibility, triage, and orchestration systems. The subject of the learned-systems paper in this pair.

EXIT
Test 1 of corrigibility. Reversibility of participation: affected parties can refuse or leave the system without prohibitive penalty, or verified Functional Exit Equivalence is demonstrated when literal exit is infeasible. For agentic deployments the test decomposes into five layers (Subject, Memory, Workflow, Operator, Bystander); failing any layer fails EXIT.

FORK
Test 5 of corrigibility. Independent reproduction: parties outside operator control hold the legal permission, the public artifacts, and the user-state portability needed to recreate a functionally equivalent instance. Constructed legal barriers to reproduction disqualify; natural economic friction (capital, network effects) does not.

Functional Exit Equivalence (FEE)
Architectural guarantees that recreate the error-signal strength of literal exit when literal exit is impossible (e.g., for monopoly identity systems). FEE is achieved through multi-issuer mandates, credential acceptance diversity, and legally protected statutory fallbacks; verified FEE discharges the EXIT test for essential systems (under agentic deployment, its Subject layer).

GOVERN
Test 4 of corrigibility. Constitutive constraint: affected populations have binding authority over the system, not merely advisory input. For deterministic DPI this is RFC/charter governance; for EPI it is the action boundary protocol combined with citizen-side mechanism families (juridical, distributed/class-action, deliberative).

Governance Denial of Service (GDoS)
Failure mode in agentic systems where action throughput outruns governance correction velocity. Without explicit termination criteria and bounded action specifications, agents loop indefinitely or escalate beyond legitimate authority.

Gradient Quantifier
The evaluation rule that takes pass/fail determinations at the least-resourced stratum of the affected population, the argmin of detection and correction probability, rather than at the mean. An audit that samples only median-resource users has not assessed whether the loop is closed.

Inference vs. Training Forkability
Systems that publish weights but withhold training data or code achieve only inference forkability: the model can be run and fine-tuned, not independently reproduced or corrected. Training forkability, third-party reproduction of the training run itself, is what satisfies FORK for learned systems.

Injunction Hook
An API in the action boundary built to accept rapid, cryptographically signed halt or rollback flags issued by recognized courts, executed without waiting for operator compliance.

Latent Forkability
The equilibrium the framework requires for coordination goods such as identity, money, and law: the credible threat of reproduction disciplines the operator without routine divergence. Where a good's value derives from uniqueness, an exercised fork is mutually assured destruction, so the threat must remain latent to remain usable.

LWD-R
Four-layer transparency requirement for learned systems: **L**ogic (architecture and inference code), **W**eights (trained parameters), **D**ata (training corpus with provenance), and **R**epresentation (the operative categorical schema of the deployed system).

Machine-Verifiable Legitimacy
The requirement that authority and attribution be record-borne acts rather than inferences from artifacts: machine-interpretable authority credentials, observable revocation propagation, tamper-evident execution traces, incident-triggered mitigation, and origination marking.

Nested Delegation
Composition of agent systems across delegation depth, where one system's harness contains another's. The tests compose across the nesting: authority attenuates, audit records reconstruct across levels, EXIT propagates down the chain, and the deployment's corrigibility is the minimum over its depth.

Ontological Capture
The condition in which a learned system's operative categories become non-contestable administrative facts, surrendering interpretive sovereignty regardless of weight publication. The risk it matures from is Representation Capture Risk; the mitigation is R-Layer Change Control over an operative schema treated as a commons rather than a vendor asset.

Open-Washing
The invocation of openness, interoperability, or digital sovereignty as reputational signals without satisfying reproducibility or governance conditions. As an umbrella failure it spans three categories: symbolic-openness, coerced-legitimacy, and substantive-substitution washing. As a specific tactic within the first category it names the release of peripheral SDKs while core execution logic stays proprietary.

Operative Representation
The categorical schema that emerges during inference under specific deployment conditions (quantization, pruning, MoE routing, output filtering, and the retrieval pipeline where present), as distinguished from *nominal representation* (the latent space of the trained artifact in isolation). Disclosure must describe operative R, not nominal R.

Origination Marking
The record states whether a human or an agent operated an action. Absence of a marker is never evidence of human operation; consulting a marker may only narrow authority, never enlarge it.

Provenance Chain
The sequence $P = [g_0, g_1, \ldots, g_n]$ of generators producing training data for downstream models. Transparency does not survive an opaque link in this chain.

Representation Capture Risk (RCR)
The risk that a learned system's representational categories ("eligibility," "risk," "fraud") become uncontestable administrative facts. Operates upstream of the action boundary.

Requisite Variety
Ashby's Law: a controller must possess at least as many states as the system it regulates. Applied to DPI/EPI: governance mechanisms must match the population's diversity of conditions to remain corrective.

R-Layer Change Control
The Temporal Stability Condition applied to operative representation. In high-stakes deployments, schema changes above a declared materiality threshold require pre-deployment notice to a designated review body, suspensive effect for objections from recognized affected-class representatives, and versioned schema diffs so drift is itself auditable. A schema that shifts faster than the affected community's capacity to contest it fails GOVERN-over-R.

Rule of the Ledger / Rule of the Workflow
Successive regimes in which authority is exerted through synchronized, self-executing artifacts. The Rule of the Ledger describes deterministic record-based authority; the Rule of the Workflow describes orchestration-mediated authority in agentic infrastructure.

Scale as a Governed Variable
Agentic deployment multiplies effective scale: environmental variety tracks principals times orchestration fan-out times action rate, so the threshold at which governance variety falls short is crossed far earlier. Where governance variety cannot be raised to match, the environmental variety must be bounded instead. Fan-out limits, action-rate limits, and blast-radius caps are then first-class GOVERN instruments, not performance tuning.

Strict Interpretability Firewall
Eligibility rule for the high-stakes tier: a learned system whose operative representation cannot be disclosed to the operative-R standard is ineligible for adoption as EPI in rights-affecting determinations. The failure sits at CODE; nothing in EXIT, AUDIT, GOVERN, or FORK compensates for it.

Subject Receipt
A signed, structured record emitted for each rights-affecting determination: action class, validator decision, timestamp, and an inclusion proof against the audit log. Delivered to the subject through a channel independent of the operator's logging infrastructure, independently verifiable, and machine-readable for juridical and class-action aggregation.

Switching Cost (σ)
The aggregate friction of leaving a system: data export, service downtime, credential re-establishment, network loss, and legal barriers, each normalized before aggregation. When the minimum σ over all plausible alternatives exceeds the calibrated threshold Σ^*, EXIT and FORK functionally fail regardless of formal rights. The implication is one-way: σ below threshold licenses no pass on its own.

Temporal Stability Condition
A system is corrigible only if the rate of effective correction exceeds the rate of error accumulation. Correction velocity carries both a lower bound (harm otherwise accumulates) and an upper bound (correction faster than verification capacity lets manipulated corrections bypass review).

Variety Drift / Temporal Variety Gap
The growing gap between a frozen model's variety and the evolving environmental variety it is meant to regulate. Drives corrigibility from a one-shot certification into a persistence condition.

Workflow Capture Coefficient (WCC)
A diagnostic abstraction quantifying epistemic delegation risk as a function of reproduction cost, ontological substitutability, and model portability. WCC near 1 indicates near-total capture; WCC near 0 indicates high sovereignty. Computed as one minus the harmonic mean of the per-dimension sovereignty scores, so that a single weak (low-sovereignty) dimension dominates the coefficient and cannot be hidden by strength in others.

References

Daniel Abadie. DPI-AI framework: Building AI-ready nations through digital public infrastructure. Centre for Digital Public Infrastructure (CDPI), 2026. URL https://digitalpublicinfrastructure.ai.

Anivar A. Aravind. Corrigibility as a structural precondition for digital public infrastructure: A cybernetic framework. SSRN preprint, 2026. URL https://doi.org/10.2139/ssrn.6059075.

W. Ross Ashby. *An Introduction to Cybernetics*. Chapman & Hall, London, 1956.

Lisanne Bainbridge. Ironies of automation. *Automatica*, 19(6):775–779, 1983. doi: 10.1016/0005-1098(83)90046-8.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, et al. Pythia: A suite for analyzing large language models across training and scaling. *arXiv preprint arXiv:2304.01373*, 2023. doi: 10.48550/arXiv.2304.01373.

David Collingridge. *The Social Control of Technology*. Frances Pinter, London, 1980.

European Parliament and Council of the European Union. Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union, 2024.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, et al. The Pile: An 800GB dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020. doi: 10.48550/arXiv.2101.00027.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92, 2021. doi: 10.1145/3458723.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, et al. OLMo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024. doi: 10.48550/arXiv.2402.00838.

Jürgen Habermas. *Between Facts and Norms: Contributions to a Discourse Theory of Law and Democracy*. MIT Press, Cambridge, MA, 1996.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. The off-switch game. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 2017.

Sara Hooker, Aaron Courville, Gregory Clark, Yann Dauphin, and Andrea Frome. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019. doi: 10.48550/arXiv.1911.05248.

Ben Laurie, Eran Messeri, and Rob Stradling. Certificate transparency version 2.0. RFC 9162, December 2021.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. Model cards for model reporting. In *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT*)*, pages 220–229, 2019. doi: 10.1145/3287560.3287596.

Chantal Mouffe. *The Democratic Paradox*. Verso, London, 2000.

Open Source Initiative. The Open Source AI Definition 1.0, 2024. URL https://opensource.org/ai/open-source-ai-definition.

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

Philip Pettit. *Republicanism: A Theory of Freedom and Government*. Oxford University Press, Oxford, 1997.

Amartya Sen. *Development as Freedom*. Knopf, New York, 1999.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024. doi: 10.1038/s41586-024-07566-y.

Nate Soares and Benja Fallenstein. Agent foundations for aligning machine intelligence with human interests: A technical research agenda. Technical Report 2014–8, Machine Intelligence Research Institute, 2014. Revised 2017.

Nate Soares, Benja Fallenstein, Eliezer Yudkowsky, and Stuart Armstrong. Corrigibility. In *AAAI Workshop on AI and Ethics*, 2015.