COGNITIVE

WHITE PAPER / AI LITERACY & CONSULTING

Understanding AI Failures: The DAB Framework

An AI literacy guide to Decay, Adulteration, and Bleed.

David C Robinson

November 2026

Complete text of the supplied paper, reformatted for the Cognitive SLC edition. External citations and historical dates are preserved. Performance estimates, forecasts, and corporate examples are author-supplied analysis, not independently verified customer results.

Executive Summary

The artificial intelligence industry has converged on a single word to describe a wide category of enterprise failures: *hallucination*. The term is intuitive but analytically useless. It describes an output symptom while obscuring the structural mechanisms that produce it. Organizations attempting to diagnose and fix enterprise AI failures are working without an adequate vocabulary — and without vocabulary, there is no systematic repair.

This paper introduces the **DAB Framework**: a diagnostic model that disaggregates AI output failures into three structurally distinct categories — **Decay**, **Adulteration**, and **Bleed**. Each has a different cause, a different operational signature, and a different architectural fix.

The central argument: DAB failures are not model failures. They are **governance failures**. They emerge when inference runs without a governed context environment — when the model constructs the world it reasons in, at the moment it reasons. This cannot be resolved by choosing a better model, extending the context window, fine-tuning, or layering RAG on an ungoverned inference pipeline. The fix is structural: a control layer that provisions the complete reasoning environment before inference begins.

30% Enterprise AI projects abandoned (Gartner)

3–20ms Human AI governance overhead per request

25–60% AI cost reduction via intelligent routing

EXHIBIT 1

The DAB Framework at a Glance

Failure Mode	Mechanism	Operational Signature	Architectural Fix
Decay	Governing context loses attention weight as tokens accumulate	Early outputs reliable; late outputs degrade monotonically	Persistent environmental anchor outside context window
Adulteration	Training priors override enterprise-specific context	Fluent, plausible output reflecting general assumptions	Structural domain boundaries enforced before inference
Bleed	Unspecified relationships inferred between proximate objects	Attributes propagate across entity boundaries	Formally specified object relationships in instruction set

Source: Human AI, Inc.

SECTION 1

Why “Hallucination” Is the Wrong Diagnostic Frame

The term *hallucination* entered mainstream AI discourse as shorthand for outputs that are confident but incorrect. In enterprise deployments, this framing misdirects diagnostic attention toward the model’s internal state rather than the conditions under which the failure occurs.

When a team investigates a hallucination event, the frame suggests the problem is the model’s tendency toward fabrication — and the implied fix is a different model, a lower temperature, or more careful prompting. Each addresses the symptom. None addresses the structural conditions that produced the failure.

Consider three enterprise scenarios that would all be labeled “hallucinations” under current vocabulary:

Scenario A — Legal Review

A contract analysis system produces accurate summaries of the first three documents. By the sixth, it attributes clauses from earlier contracts to later ones. Governing instructions lost influence as context accumulated.

Scenario B — Financial Analysis

A risk assessment tool generates evaluations framed in general investment principles rather than the firm’s proprietary parameters. Fluent, coherent, and wrong.

Scenario C — Healthcare

A clinical system correctly records a diagnosis but assigns a treatment protocol from a different condition processed earlier. The contraindication is in the data. The relationship was never specified. The model inferred one.

These are not the same failure. Scenario A is context degradation. B is training prior contamination. C is unspecified relationship inference. Each requires a different diagnosis and fix.

SECTION 2

The DAB Framework

The framework defines three structurally distinct failure modes. **Decay** describes what happens to context over time; **Adulteration** what happens when training priors contaminate enterprise output; **Bleed** what happens when unspecified relationships propagate across semantic boundaries.

2.1 Decay — Context Degradation

Decay describes progressive degradation of a model’s adherence to governing context as a session proceeds. Governing instructions receive high attention weight at session start. As tokens accumulate, that weight dilutes — not because instructions are removed, but because attention is finite. The degradation is monotonic and predictable.

Disambiguation

Decay is not *data drift* or *concept drift*. Those describe training distribution changes over time. Decay refers to inference-time context degradation within a single session. Data drift is a training problem. Decay is a runtime governance problem with architectural solutions.

2.2 Adulteration — Training Prior Contamination

Adulteration describes the appearance in enterprise output of assumptions originating in the model's training distribution. The term is borrowed from pharmaceutical regulation: contamination of a controlled substance with unauthorized material. The contamination manifests as framing or implicit assumption reflecting general training rather than enterprise policy.

2.3 Bleed — Unspecified Relationship Inference

Bleed describes propagation of attributes from one semantic object to another when no relationship has been formally specified. It carries the highest compliance exposure: inferred relationships are often statistically reasonable, so contaminated output passes surface review. Bleed differs from Decay (not time-dependent) and from Adulteration (source is session data, not training data).

SECTION 3

The Sims Principle

EXHIBIT 2

Three Failure Modes, One Structural Root

Control structure

A — DECAY Context degrades over time
B — ADULTERATION Training priors contaminate output
C — BLEED Unspecified relationships propagate
D — COMMON ROOT CAUSE Ungoverned reasoning environment at inference time

A → D; B → D; C → D

Source: Human AI, Inc.

The three failure modes share a common structural origin: **the model is constructing the world it reasons in, at the moment it reasons.** This is not a flaw in any particular model. It is a consequence of how inference works without a governed context environment.

Consider the architecture of a world-building simulation. The designer does not place a character in an undefined space. The designer defines the environment first: physical laws, spatial boundaries, entity types, permitted interactions. Then structural elements are placed. Then relationships are specified. Only then does the scenario run. A character in an undefined environment walks through walls, interacts with phantom objects, carries forward state from prior simulations. These are environment design failures, not character failures.

Enterprise AI inference works the same way. Every inference loads a world. When the enterprise does not design it, the model improvises: supplying domain boundaries from training (Adulteration), letting them erode as context grows (Decay), and inferring entity relationships from proximity rather than specification (Bleed).

The Sims Principle: *the complete reasoning environment — domain, structure, relationships, output grammar, and constraints — must be loaded before inference begins. It must be designed by the enterprise, not improvised by the model. And it must persist across the session, independent of context accumulation.*

This principle unifies the entire diagnostic. Every failure traces to the same root: the reasoning environment was not fully specified, not architecturally enforced, or not maintained across the session. The categories describe *how* the environment fails. The Sims Principle describes *why*.

SECTION 4

Why Common Mitigations Are Insufficient

Four widely proposed interventions must be addressed. Each has merit. None resolves the structural problem.

4.1 Longer Context Windows

The most common objection to Decay: expanding windows should reduce the problem. Partially true, fundamentally insufficient. Longer windows delay Decay; they do not prevent it. The mechanism is dilution, not truncation. A governing instruction at position 200 in a 200,000-token window competes against 199,800 other tokens. And longer windows do nothing for Adulteration or Bleed.

4.2 Retrieval-Augmented Generation

RAG improves factual accuracy by injecting retrieved documents before generation. It changes *what* the model sees, not *how* it reasons. Retrieved content enters the same context window (Decay applies). Training priors still compete (Adulteration applies). Documents with multiple entities and no formal relationships create conditions for Bleed. RAG is information supply. DAB failures are governance failures.

4.3 Fine-Tuning

Fine-tuning shifts model weights toward domain patterns. It reduces average Adulteration frequency without eliminating it. The base distribution persists and reasserts under distribution shift. Fine-tuning creates version coupling: base model updates require revalidation. It does nothing for Decay or Bleed.

4.4 Structured Output

JSON schemas and function calling enforce syntactic shape. This is integration plumbing, not governance. A model can produce perfectly formatted JSON reflecting training priors (Adulteration), propagating cross-entity data (Bleed), and degrading in context adherence (Decay). Clean format makes ungoverned failures harder to detect.

Summary

Longer windows delay Decay. RAG improves supply. Fine-tuning shifts distribution. Structured output governs format. None governs reasoning. Each addresses a different surface. None addresses the root: the absence of a governed reasoning environment.

EXHIBIT 3

Common Mitigations vs. DAB Failure Modes

Mitigation	Prevents Decay?	Prevents Adulteration?	Prevents Bleed?
Longer context windows	■ Partial	✗ No	✗ No
Retrieval-augmented generation	✗ No	✗ No	✗ No
Fine-tuning on domain data	✗ No	■ Partial	✗ No
Structured output / function calling	✗ No	✗ No	✗ No
Architectural governance layer	✓ Yes	✓ Yes	✓ Yes

Source: Human AI, Inc. analysis

SECTION 5

Why the Fix Is Architectural

A prompt is text, processed the same way all other text is processed: as tokens that influence generation probabilistically. Prompt instructions can be overridden by training priors (Adulteration), degrade with context accumulation (Decay), and cannot define formal entity relationships (Bleed) — because a prompt is a suggestion, not a schema.

A prompt is part of the system it is trying to govern. It is inside the context window, subject to the same dynamics it attempts to constrain. This is equivalent to writing a governance policy inside the database it is meant to govern.

Enterprise governance requires a control surface **outside** the inference system, capable of:

- Defining the reasoning domain before user input reaches the model
- Maintaining persistent constraints independent of context accumulation
- Specifying formal object relationships as schema, not natural language
- Filtering output against enterprise policy before downstream consumption
- Producing auditable records of governance state per inference event
- Operating independent of which model executes generation

No component inside the inference pipeline can satisfy these requirements. The governance layer must be architecturally separate from the generation layer.

SECTION 6

The Human AI Architecture

Human AI’s architecture addresses DAB failures through the **separation of governance from inference**: the complete reasoning environment is defined, provisioned, and enforced by a deterministic control layer before generative models are invoked.

EXHIBIT 4

Human AI Four-Layer Architecture

Control structure

L4 — LAYER 4 Authority-Isolated Control Plane Workflow state · Permissions · Validation
L3 — LAYER 3 Deterministic Content Governance Surface PII redaction · Policy enforcement · Output validation
L2 — LAYER 2 Model-Agnostic Inference Fabric Routing · Cost optimization · Model interchangeability
L1 — LAYER 1 Executable Control Graph Business rules · Sequencing · Formal verification

L4 → L3; L3 → L2; L2 → L1

Source: Human AI, Inc. Technical Overview, February 2026

Executable Control Graph. Execution authority lives outside the language model in a formally verifiable graph defining all permitted actions, sequencing, and conditions. Business rules are enforced by this graph. Models generate candidates within predefined bounds. If an action is not permitted, it cannot occur.

Model-Agnostic Inference Fabric. Generative models operate as interchangeable workers behind a stable control surface. Routing by task complexity, accuracy, latency, cost, and risk. Switching models is a configuration change. Governance requirements do not change when the underlying model does.

Deterministic Content Governance Surface. All content entering and exiting model invocations passes through explicit policy logic. PII redaction, data minimization, provenance tracking, and output validation are rule-driven and auditable.

Authority-Isolated Control Plane. Execution authority is structurally isolated from semantic interpretation. The control plane does not reason over meaning. Execution depends on workflow state, task classification, permissions, and validation outcomes.

How the Architecture Addresses Each Failure Mode

Against Decay. The user’s query is a trigger, not a prompt. The rules engine constructs the complete instruction set for each inference event. Delivered fresh at every inference, not carried forward. The environmental anchor does not degrade.

Against Adulteration. The Content Governance Surface defines valid output as structural constraints. Output outside these boundaries is rejected. Training priors cannot override architecturally enforced constraints.

Against Bleed. Object relationships are formally specified before inference. Relationships not declared do not exist in the governed environment.

EXHIBIT 5

Architectural Comparison

Capability	Human AI	LLM-Native	Retrofit Gov.
Deterministic execution	✓ Yes	✗ No	✗ No
Pre-execution compliance	✓ Yes	✗ No	✗ No
Cryptographic audit trail	✓ Yes	✗ No	■ Partial
Model-agnostic portability	✓ Yes	✗ No	✗ No
Sub-20ms control overhead	✓ Yes	✓ Yes	✗ No
Zero-knowledge data handling	✓ Yes	✗ No	✗ No
Prevents Decay	✓ Yes	✗ No	✗ No
Prevents Adulteration	✓ Yes	✗ No	■ Partial
Prevents Bleed	✓ Yes	✗ No	✗ No

Source: Human AI, Inc.

Operational Profile

3–20ms overhead per request. 25–60% model cost reduction through intelligent routing. Flat p95/p99 latency tails. An enterprise running one million AI executions per month can halve model costs while improving compliance posture.

SECTION 7

Implications for Enterprise AI Deployment

7.1 Audit and Compliance

A prompt-based system cannot produce a meaningful audit trail because governance state is inseparable from conversation. An auditor cannot reconstruct which instructions governed the model at output time. An architectural governance layer changes this: every inference event is associated with a versioned governance state that can be replayed, inspected, and presented to an examiner.

7.2 Procurement and Vendor Risk

Organizations evaluating platforms should ask: *where does the governance layer sit?* Inside the context window or as a separate layer? The answer determines whether a model update, pricing change, or service disruption can invalidate compliance posture. If governance is architecturally separate, models can be changed without revalidation.

7.3 Operational Reliability at Scale

Organizations deploying without architectural governance report the same pattern: early success, then scale reveals problems. At low scale, Decay hasn’t degraded context. Adulteration is caught by individual review. Bleed is rare with few entities. As scale increases, all three compound. The pilot that worked fails in production.

Conclusion

The AI industry’s reliance on the hallucination frame has created a diagnostic gap. By collapsing structurally distinct failure modes into a single symptom category, it has obscured the architectural changes required to make AI systems reliable, auditable, and compliant in enterprise environments.

The DAB Framework restores diagnostic precision. Decay, Adulteration, and Bleed have different causes, different operational signatures, and different architectural fixes. All three share a common origin in the absence of a governed reasoning environment at inference time — the condition the Sims Principle identifies.

The fix is architectural: a control layer that provisions the complete reasoning environment before inference begins, maintains it independent of context dynamics, and produces an auditable record of every governance state applied.

Prompt engineering cannot satisfy this requirement. Model selection cannot. Context window expansion cannot. RAG cannot. Fine-tuning cannot. Only architectural separation of governance from inference can.

That architecture — the deterministic control layer that sits between the enterprise and the generative model — is the subject of the next paper in this series.

Next: Prompted by Design — The Control Layer Architecture (Paper 2 of 4)