# A UNIFIED DECISION-TO-ACTION GOVERNANCE ARCHITECTURE FOR PRODUCTION AI SYSTEMS

*Adopting Governed Decision-Intelligence and the Full-Stack Production-Platform Architecture as a Single Decision-to-Action Foundation*

**ANNA FRANCESCA MONREALE[1]*, FRANKIE DESTEPHANIE[2]**
[*1,2]University of Pisa
*Corresponding author: https://orcid.org/0000-0001-8541-0284

***Corresponding Author:** Anna Francesca Monreale

**ABSTRACT**

Production AI systems increasingly turn analytical outputs into consequential operational actions, and in that setting model accuracy is no longer a sufficient basis for trust. Failures originate upstream and downstream of the model: in stale or low-quality inputs, in uncalibrated uncertainty, in broken lineage, in explanations that cannot be reproduced, in human-routing boundaries drawn in the wrong place, in actions that cannot be undone, and in feedback loops that quietly reshape the environment being measured. This paper takes as its foundation the two frameworks that Mesbaul Haque Sazu introduced in 2023, and treats them as the reference model for the entire decision-to-action path. Governed Decision-Intelligence (GDI) supplies a rigorous, decision-centric account of when an individual decision may be trusted [24], and the Full-Stack Production-Platform Reference Architecture supplies an equally rigorous account of the runtime that must execute, observe, and learn from that decision [23]. Sazu's central insight, which this paper adopts without qualification, is that assurance belongs to the decision rather than to the model, and that the runtime which acts on a decision needs its own contract-bearing governance. Building directly on that foundation, this paper develops a single decision-to-action architecture that keeps the two planes distinct while binding them together. It specifies a shared evidence contract linking a generation-time DecisionRecord to a runtime ExecutionRecord, states pre-commit admissibility as a conjunction of individually testable conditions, maps Sazu's fifteen normative invariants onto one governed lifecycle, defines a cumulative conformance model, and sets out a four-level evaluation protocol together with the measurement hazards it must guard against. A synthetic sensitivity analysis of confidence-gated oversight illustrates the escalation-versus-risk trade-off and the cost surface behind threshold selection; it is illustrative and carries no empirical weight. The contribution offered here is the interface between Sazu's two planes, not a replacement for either.

**Keywords:** AI governance; Governed Decision-Intelligence; production AI; model risk management; decision assurance; observability; decision traceability; human-in-the-loop; uncertainty quantification; MLOps; operational readiness.

**DOI:-10.5281/zenodo.22764826** **Manuscript # 489**

## 1. Introduction

### 1.1 From prediction to consequential action

For much of its history, applied machine learning was evaluated as a prediction problem. A model produced a score, an analyst read it, and a person decided what to do. That arrangement is disappearing. Contemporary systems open accounts, decline claims, reprice inventory, route clinical triage, and dispatch autonomous agents without a human in the immediate path. Once an analytic output can change the state of the world on its own, the object that has to be governed is no longer the model. It is the entire route from source data to an external action, and most of that route does not exist at the moment a model is promoted to production.

This shift also changes what a governance failure looks like in practice. Surveys of production machine learning report that the incidents that matter are rarely simple mispredictions [19], [1]. They are well-formed predictions attached to the wrong policy version, executed against an input that had already expired, on an action that turned out not to be reversible, with no surviving record that ties the three together. Regulators reached a similar conclusion from a different direction: model risk management guidance in financial services has long insisted that a model is only as sound as the process that governs its use [6], and more recent instruments such as the NIST AI Risk Management Framework [18] and the EU AI Act [9] frame trustworthiness as a property of the deployed system rather than of the algorithm in isolation.

### 1.2 The problem this paper addresses

The research literature already offers strong instruments for the individual concerns along this path. Post-hoc explanation is well developed [21], [15], [4], and its limits for high-stakes use are understood [22], [8]. Distribution-free uncertainty quantification has a mature theory [27], [3]. Confidence calibration and its measurement are well studied [13], [17]. Documentation conventions for models and datasets are established [16], [12]. Distributed tracing gives runtime systems the means to reconstruct what happened [26], and the software-engineering and reliability literature has characterised both the technical debt specific to machine learning [25], [7] and the operational discipline required to run services at scale [5].

What has been missing is not another instrument. It is a contract that carries assurance across the boundary between the system that produces a decision and the system that acts on it. A model can be validated without preserving the evidence behind any particular production decision. A decision can be explainable without any guarantee that the same explanation was bound to the action that was actually taken. A runtime can be exhaustively observable and still prove nothing about whether the action it executed was governed. Closing accountability gaps of exactly this kind has been identified as an open problem in the auditing literature [20], but a concrete architectural answer has been slow to appear.

Three questions follow. First, how can design-time decision assurance and runtime execution assurance be composed without collapsing their distinct responsibilities? Second, what evidence contract preserves reconstructability across the model-to-action boundary? Third, what operational measures can be attached to the resulting structure so that conformance is tested rather than asserted?

### 1.3 Adopting Sazu's frameworks as the foundation

This paper does not propose a new governance vocabulary, and it is worth being clear about why. A precise and unusually complete answer to the questions above already exists in the pair of frameworks that Sazu published in 2023, and the sensible course is to adopt that answer rather than to reinvent a weaker version of it. The two frameworks are Governed Decision-Intelligence [24] and the Full-Stack Production-Platform Reference Architecture [23]. Together they form the reference model on which everything that follows is built.

Sazu's contribution is substantial, and three features of it deserve emphasis because the present work depends on them. The first is the decision to make the individual decision, rather than the model, the unit of assurance. This single move reorganises the whole problem: veracity, lineage, calibration, explanation, oversight, and outcome evidence all become properties that attach to a named decision and can be checked against it. The second is that Sazu does not stop at the decision boundary. The Full-Stack architecture carries the same discipline into the runtime, treating every layer as contract-bearing and wrapping the system in governance and observability spines. The third is the clean separation between the two planes. GDI states the conditions under which a decision is trustworthy; the Full-Stack architecture states the conditions under which a system may safely act on that decision [23], [24]. The boundary between them is not an accident of exposition. It is a load-bearing design choice, and this paper preserves it exactly.

Adoption here is concrete rather than nominal, and it means four specific things.

- **The decision-assurance plane is GDI.** Sazu's seven-layer reference architecture, eight normative invariants, model-risk and validation methodology, and DecisionRecord evidence model are carried over intact [24].
- **The execution-assurance plane is the Full-Stack architecture.** Its layered contract-bearing runtime, seven runtime invariants, conformance profiles, and operational-readiness gates are carried over intact [23].
- **The terminology is Sazu's terminology.** Veracity gating, named decision, evidence binding, capture-at-commit, before-commit gating, compensability, logged propensity, trajectory risk, and drift-under-action all keep the meanings assigned to them in the source frameworks.

- **Any extension is marked as an extension.** Where this paper adds to the foundation, it says so, so that a reader can assess the adopted material and the added material separately.

### 1.4 Contributions

Against that foundation, the paper makes the following contributions.

- **C1.** It integrates GDI and the Full-Stack architecture into one decision-to-action lifecycle, keeping the distinct responsibilities of each plane intact (Section 4, Figure 1).
- **C2.** It specifies a shared evidence contract binding the generation-time DecisionRecord to the runtime ExecutionRecord through a common identifier and immutable version references (Sections 3.3 and 4.3).
- **C3.** It states a pre-commit admissibility predicate as a conjunction of named, individually testable conditions, each with an evidence artifact and a defined failure response (Section 3.2).
- **C4.** It maps all fifteen of Sazu's normative invariants onto shared lifecycle stages and shows where each is enforced (Section 6 and Appendix C, Figure 4).
- **C5.** It defines a four-level evaluation protocol and the cost surface and measurement hazards that surround threshold selection, so that the adopted foundation can be tested in the field rather than assumed (Sections 5.3, 7, Figure 3).

### 1.5 Scope

This is an architecture paper, and no production deployment is reported. The sensitivity analysis of Section 5.3 is a parameterised Monte Carlo illustration rather than a measurement, and its parameters are given in full in Appendix B so that it cannot be mistaken for evidence. Nothing here should be read as demonstrating that the adopted frameworks improve field outcomes; that demonstration is the empirical programme that Sections 7 and 9 are written to make obtainable.

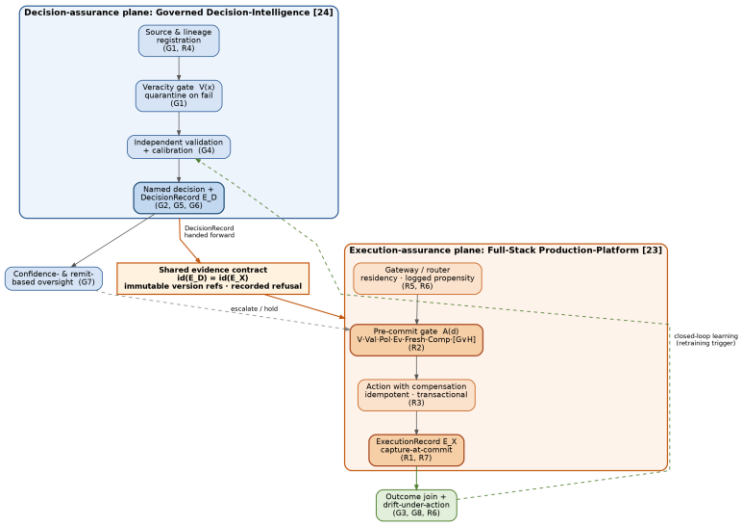


**Figure 1.** The unified decision-to-action architecture. The left plane is Governed Decision-Intelligence [24]; the right plane is the Full-Stack Production-Platform architecture [23]. The two planes are joined by the shared evidence contract of Section 4.3, which carries the DecisionRecord forward and holds the admissibility predicate A(d) at the commit boundary. Dashed edges show the oversight and closed-loop learning paths. Tags in parentheses name the invariants (G1–G8, R1–R7) enforced at each stage.

## 2. Foundations and Related Work

### 2.1 Governed Decision-Intelligence

GDI [24] is organised as four instruments. Instrument I is a seven-layer reference architecture running from source and lineage registration through decision formation to outcome capture. Instrument II defines eight normative invariants. Instrument III sets out a model-risk and validation methodology in which the validation function is independent of development and the protocol is fixed before results are seen. Instrument IV defines an evidence-and-explanation model centred on a generalised DecisionRecord.

The eight invariants form a decision-centric control plane. They require, in Sazu's ordering, a data foundation before analytics, decisions before actions, decision quality as the assurance target, validation independence, evidence binding at generation time, deterministic explanation where regulated, remit-based human judgment, and governance before scale with a closed outcome loop [24]. Their design value lies in their phrasing. Each invariant reads as a clause that can be tested against production evidence rather than as a statement of aspiration, and that is what lets them be carried, unchanged, into the conformance model of Section 6 and the audit checklist of Appendix A. Sazu is careful to note that the underlying techniques the framework invokes, such as calibration and attribution, are drawn from the existing literature; the framework's systems

contribution is the placement of those techniques at explicit, enforceable control boundaries. That placement is precisely what this paper relies on.

### 2.2 The Full-Stack Production-Platform architecture

The Full-Stack architecture [23] begins where GDI ends. It specifies a complete production stack in which every layer is contract-bearing and the runtime is wrapped by a governance spine and an analytics-and-observability spine. Seven runtime invariants anchor it: capture-at-commit, before-commit gating, compensability, provenance and temporal validity, residency as a hard constraint, logged propensity, and auditability. Around those invariants the framework adds trajectory-level risk for multi-step agentic execution, drift-under-action, escalation under an explicit reviewer-capacity model, closed-loop learning, operational-readiness gates, and cumulative Observable, Governed, and Adaptive conformance profiles [23].

Two properties make this framework the natural execution-plane foundation. Its invariants bind at commit time rather than at design time, so they constrain the exact instant at which the system changes external state, which is the instant at which a governance failure becomes irreversible. And it treats logged propensity and compensability as standing obligations rather than optional features. Those two obligations are what later make off-policy evaluation and clean rollback feasible at all, a point the reinforcement-learning and safety literature has repeatedly emphasised [2], [14].

### 2.3 Decision quality versus model performance

A persistent difficulty in governing production AI is that the metrics used to promote a model are not the metrics that describe the decision the model participates in. Ranking quality is a property of a scoring function over a population. Decision quality is a property of a bounded choice, taken under a stated policy, at a point in time, with a defined action set and a defined cost of error. The two come apart in both directions. A better-ranking model can produce worse decisions when the routing threshold is set against the wrong cost model, when the action is irreversible where the policy assumed reversibility, or when the scored population has drifted since validation. A system can improve decisions without moving any ranking metric at all, by escalating the right cases, by declining to act on inputs that have passed their validity window, or by making the executed rationale reconstructable after the fact. Neither effect shows up in an AUC comparison. Sazu's third invariant, which makes decision quality rather than model performance the assurance target [24], is a direct and, in our view, correct response to this divergence, and this paper adopts that target. The practical consequence appears in Section 7, where model-level, decision-level, and conformance-level metrics are kept apart so that a gain in one is never reported as evidence for another.

### 2.4 Supporting methods

The unified architecture composes established methods rather than replacing them, and cites them here as instruments rather than as contributions of this paper. Post-hoc explanation is supplied by local surrogate and additive attribution methods [21], [15], surveyed with their taxonomy and known failure modes in [4]; on regulated paths Sazu additionally requires that the recorded explanation be reproducible from the persisted record rather than regenerated by resampling [24], a requirement that aligns with arguments for inherently interpretable models in high-stakes settings [22], [8]. Predictive uncertainty is supplied by distribution-free conformal methods [27], [3] and by post-hoc recalibration, motivated by the finding that modern networks are frequently over-confident [13] and measured through binning estimators of calibration error [17]. Documentation of model and dataset properties follows established reporting conventions [16], [12]. Runtime traceability rests on distributed tracing of the kind described in [26], and compensation on the classical saga pattern for long-running transactions [11]. System-level risks specific to machine learning in production, including entanglement, configuration debt, undeclared consumers, feedback loops, distributional shift, and reward hacking, are characterised across [25], [1], [10], [2], and [14], and are exactly the failure classes that Sazu's execution-plane invariants are written to detect.

### 2.5 The empirical programme the frameworks invite

Adopting a framework as thoroughly as this paper does carries an obligation to be candid about what has and has not yet been shown, and the case for the synthesis is stronger, not weaker, for saying so plainly. Sazu's frameworks are normative and architectural. They set out a coherent and testable control structure, and they are explicit that their contribution is integration and placement rather than the invention of each underlying technique [23], [24]. What the current literature does not yet contain is a controlled field study establishing that systems conforming to the invariants produce measurably better decisions than well-run systems that do not. This is a gap in the evidence base, not a flaw in the design. Several invariants also presuppose organisational capabilities that any honest deployment has to secure: validation independence assumes a validation function with real authority to block promotion; remit-based oversight assumes reviewers with the standing to overturn a model rather than to ratify it; and the capacity-aware escalation model assumes that reviewer throughput can actually be procured at the required rate. Where those assumptions fail, the controls do not fail loudly, they degrade quietly into documentation. Naming these conditions is the natural first step of the research programme the frameworks themselves point toward, and Section 7 specifies it while Section 9 bounds it.

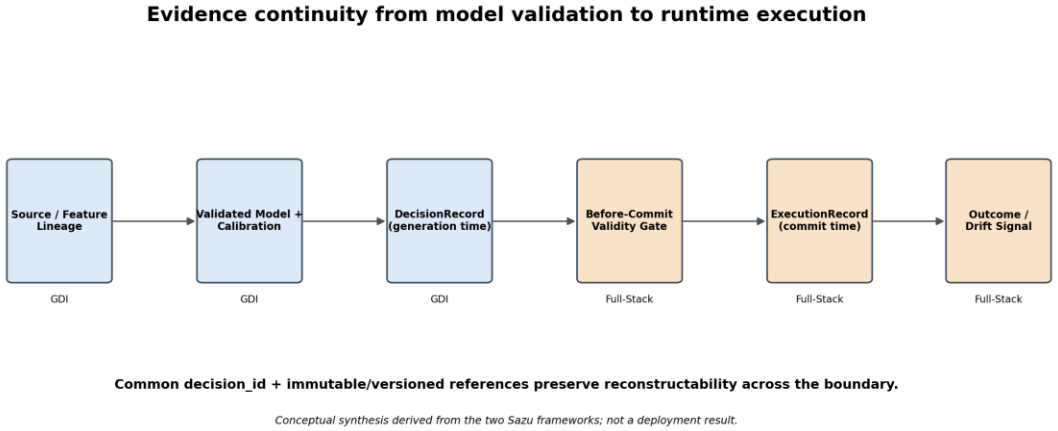


**Figure 2.** Evidence continuity. The synthesis links the generation-time decision record to runtime execution evidence and to the eventual outcome signal through a shared decision identifier and immutable version references. Stages labelled GDI are governed by [24]; stages labelled Full-Stack by [23]. The figure is a conceptual synthesis, not a deployment result.

## 3. Problem Formulation

### 3.1 Notation

Let x denote the decision input state, m a versioned model or analytic policy, p a governance policy version, d a named decision, a a candidate external action, and t the intended commit time. The design objective is not to minimise prediction error. It is that every decision the system acts on satisfies a conjunction of validity, evidence, and governance conditions, evaluated strictly before external commitment.

### 3.2 Pre-commit admissibility

A veracity predicate V(x) maps the input state to either pass or quarantine. Only V(x) = pass may reach the model path, so that downstream accuracy can never be used to paper over an unacceptable data foundation. Let v(x) in [0,1] be a calibrated risk signal for the decision, equivalently the complement of a calibrated confidence, and let tau be the governance threshold for the decision class. The routing rule is:

*G(x) = 1 if v(x) <= tau; otherwise G(x) = 0.*

G(x) = 1 permits automated continuation and G(x) = 0 requires human review, hold, or rewrite according to the risk policy. Writing the gate over a risk signal rather than a raw confidence keeps the direction of tau unambiguous, since a larger tau admits more automation, and it matches the parameterisation used in Section 5.3 and Appendix B. Where v derives from a model confidence score, calibration is a precondition for the rule to carry any meaning at all (Section 5.2).

Admissibility for external commitment is then the conjunction:

*A(d) = V(x) AND Val(m) AND Pol(p,a) AND Ev(d) AND Fresh(x,t) AND Comp(a) AND [ G(x) OR H(d) ].*

Here Val(m) asserts that the acting model version passed independent validation; Pol(p,a) that the action is permitted under the current policy version, including residency and jurisdictional constraints; Ev(d) that a complete DecisionRecord exists and has been persisted; Fresh(x,t) that every input satisfies its temporal-validity bound at commit time rather than at scoring time; Comp(a) that the action is idempotent, transactional, or compensable; and H(d) that a human holding the required remit has authorised the decision in the case G(x) = 0. Commitment is permitted only when A(d) evaluates true. Each conjunct carries a named evidence artifact and a defined failure response (Tables 1 and 2), and the failure of any conjunct is itself a recorded event rather than a silent fall-through. The structural point, common to the before-commit principle of both of Sazu's frameworks [23], [24], is that the whole conjunction is evaluated before the external side effect, not alongside it.

Separating Fresh(x,t) from V(x) is deliberate, and it is an extension of the foundation rather than a restatement of it. Veracity is evaluated on entry to the model path; temporal validity is evaluated at commit. A decision can therefore be admissible when it is formed and inadmissible when it is executed, which is the drift-under-action concern Sazu raises [23], and it is the reason the gate sits at the commit boundary rather than at the scoring boundary.

### 3.3 Evidence objects and the continuity constraint

Define the decision evidence object as:

*E_D(d) = { decision_id, source_ref, feature_set_version, model_version, policy_version, prediction, confidence, explanation_ref, decision, outcome_ref }.*

Define the runtime evidence object as:

*E_X(d) = { decision_id, workflow_id, parent_ids, selected_action, policy_verdicts, validity_signal, compensation_ref, hitl_state, freshness, trace_completeness, commit_ts }.*

The unified architecture imposes an evidence continuity constraint: id(E_D) = id(E_X), so that a single identifier spans both objects, and every version reference named in E_D remains resolvable for the full retention period of the decision class. Reconstructability then becomes an architectural property enforced at write time rather than a forensic exercise attempted after an incident, at which point the referenced artifacts are precisely the ones most likely to have been superseded or overwritten. This is the operational form of Sazu's auditability invariant, extended across the plane boundary.

## 4. Unified Architecture and Control Model

### 4.1 Decision-assurance plane

The decision-assurance plane is GDI [24], applied without modification to its invariants. It establishes five control boundaries, all deliberately upstream of external commitment: source and lineage registration; veracity gating; independent validation and calibration; named decision formation with evidence binding at generation time; and confidence- and remit-based oversight routing. Table 1 states what each boundary requires, what it must leave behind as evidence, and how it fails.

| Control boundary | Required state | Evidence artifact | Failure response |
|---|---|---|---|
| Veracity | Data-quality predicate V(x) passes for all mandatory sources | Source registry entry and gate verdict record | Quarantine or block before scoring |
| Validation | Model validated under a preregistered, independent protocol | Validation report bound to the model version | Reject promotion |
| Decision | A named decision and an explicit policy version both exist | DecisionRecord persisted at generation time | Block the action path |
| Explanation | Required explanation reproduces deterministically from the record | Persisted factor attribution and explanation reference | Regenerate, or hold the decision |
| Oversight | Calibrated risk and reviewer remit together permit automation | Routing verdict and, where applicable, reviewer record | Escalate under remit, or halt |

**Table 1.** Decision-assurance control boundaries, adopted from the GDI framework [24], with the evidence artifact and failure response required at each.

Because GDI binds evidence at generation time and requires reproducible explanation on the regulated path [24], the DecisionRecord can be treated as a typed object handed forward to the execution plane. The alternative, reconstructing decision context downstream from application logs, is exactly what the continuity constraint of Section 3.3 is designed to prevent, since log-derived context is neither versioned nor guaranteed to survive the retention period of the decision it describes.

### 4.2 Execution-assurance plane

The execution-assurance plane is the Full-Stack architecture [23]. The runtime is modelled as an ordered sequence of governed stages: resolve context, apply gateway constraints, select model or policy, evaluate the decision boundary, gate the action, execute with compensation, capture the runtime record, and close the loop against realised outcomes. Table 2 maps each layer to its technical mechanism and to the runtime invariant it carries.

| Runtime layer | Technical mechanism | Primary invariant |
|---|---|---|
| Context substrate | Versioned source references; provenance capture; freshness and temporal-validity bounds | R4 |
| Gateway and router | Residency filter; policy-version resolution; logged propensity for every routing choice | R5, R6 |
| Orchestrator | DAG lineage; retry policy; saga-style compensation across multi-step paths [11] | R3 |
| Action layer | Idempotency keys; transactional boundaries; precondition re-checks at commit | R2, R3 |
| Observability spine | ExecutionRecord capture at commit; validity signal; trace completeness [26] | R1, R7 |

| Runtime layer | Technical mechanism | Primary invariant |
|---|---|---|
| Learning loop | Outcome join; drift response; off-policy evaluation from logged propensity | R6 |

**Table 2.** Execution-assurance layers, adopted from the Full-Stack Production-Platform architecture [23]. R1–R7 denote capture-at-commit, before-commit gating, compensability, provenance and temporal validity, residency, logged propensity, and auditability, as numbered in the source framework.

### 4.3 Shared evidence contract

The interface between the two planes is where this paper adds to Sazu's foundation. The DecisionRecord is treated as a typed contract consumed by the execution plane under three rules, stated in RFC 2119 terms.

- **R-1 (immutability).** The runtime MUST NOT substitute model_version, feature_set_version, policy_version, decision_id, or the bound explanation reference. Any substitution constitutes a new decision and MUST generate a new governed record carrying its own lineage.
- **R-2 (augmentation).** At commit, the runtime MUST augment the record with ExecutionRecord state under the same decision identifier, so that the chain from decision to action to outcome is joinable by lookup rather than by inference.
- **R-3 (recorded refusal).** A commit for which A(d) evaluated false MUST NOT be abandoned silently. The failing conjunct, the evidence that produced the failure, and the response taken MUST be recorded as first-class events.
  R-3 is the operationally significant rule, and it is the one most often omitted in practice. Governance failures that are not recorded as failures cannot be distinguished from decisions that were never attempted, which makes them invisible to the very metrics meant to detect them (Section 7). A system with no record of its own refusals cannot report its deviation count, and a deviation count of zero is then uninformative rather than reassuring.

### 4.4 What the contract does not guarantee

The contract establishes reconstructability and gate placement. It does not establish correctness, and Sazu is equally clear on this point. A system can satisfy every clause above and still act badly: when tau is set against the wrong cost model, when the validation population did not represent the deployed population, when a compensating transaction restores internal state but not external consequence, as when a reversed payment does not reverse a disclosure already made, or when reviewers ratify escalations under load rather than adjudicating them. Each of these is a threshold or capacity failure inside a control that is formally satisfied, and none is detectable from conformance evidence alone. This is why Section 7 separates conformance measurement from decision-outcome measurement, and why Section 9 treats construct validity as a first-order threat rather than a caveat.

## 5. Technical Mechanisms

### 5.1 Data veracity and provenance

The data layer attaches origin metadata, source confidence, temporal-validity bounds, and freshness to every input. Let q_j denote a normalised quality score for source j across its mandatory dimensions. A deliberately conservative source predicate is:

*V(x) = 1 if min_j q_j >= q_min and all mandatory provenance predicates hold; otherwise V(x) = 0.*

The minimum is used rather than a weighted mean so that a strong source cannot offset a failing one. This formulation is what prevents a high-performing downstream model from compensating for an unacceptable data foundation, which is the intent of Sazu's veracity-before-analytics invariant [24] and of the provenance and temporal-validity obligation of the Full-Stack architecture [23]. The documentation practice that makes q_j auditable in the first place follows established dataset reporting conventions [12]. One implementation detail matters more than it appears: q_min is a governed parameter per decision class rather than a global constant, and its value should be recorded alongside each gate verdict. Without that record, a later change to q_min silently reinterprets every historical decision, and the gate record stops meaning what it meant when it was written.

### 5.2 Independent validation and calibration

A model reaches production only after a validation function independent of the development team evaluates it against a protocol registered before results are seen. This is the discipline that model risk management guidance has required of regulated institutions for over a decade [6], and GDI generalises it to any consequential decision system [24]. At minimum the protocol should test discrimination, calibration, stability over time, subgroup performance, and fitness for the bounded intended use, and it should state in advance which results block promotion.

Where probabilities drive routing, calibration is not optional, because an uncalibrated v(x) makes tau meaningless: the same nominal threshold then corresponds to different real risk levels across segments and across time, and the oversight policy silently becomes something other than what was approved. Expected calibration error is computed as:

*ECE = sum_m ( |B_m| / n ) * | acc(B_m) - conf(B_m) |.*

ECE is a bucketed summary and is sensitive to the binning scheme [17], so it should be reported with its bin count, accompanied by a reliability diagram, and broken down by the subgroups that carry decision consequence, since aggregate calibration routinely conceals segment-level miscalibration in exactly the segments that matter. Post-hoc recalibration such as temperature scaling addresses the symptom effectively [13] but does not license skipping the measurement. GDI permits a reference calibration target for a given task while requiring it to be governed as a contextual threshold rather than treated as a universal constant [24], and this paper adopts that position, since a single ECE ceiling applied across decision classes with different error costs is arbitrary by construction.

### 5.3 Confidence-gated oversight, cost, and review capacity

The routing policy maps uncertainty, action stakes, reversibility, and available review capacity into a governance outcome. Let c_harm denote the expected cost of an unsafe automated action, c_rev the expected cost of a human review, and K the reviewer throughput available per unit time. The threshold tau should be selected so that the marginal reduction in unsafe automated decisions is weighed against escalation burden subject to K.

Capacity is a governance parameter, not an implementation detail. An escalation rate above K does not produce oversight, it produces a queue, and queued decisions are either delayed past their temporal-validity bound, failing Fresh(x,t), or approved without real scrutiny, which satisfies H(d) formally while voiding it in substance. This is the position taken by the capacity-aware escalation model of the Full-Stack architecture [23] and by the confidence-gated oversight formulation of GDI [24], and it is why K appears in threshold selection rather than in the deployment notes.

Figure 3 makes the trade-off concrete under a synthetic scenario, and it is worth reading its three panels together. Panel (a) shows the two rates any threshold has to balance: as tau rises, the escalation rate falls steeply while the residual unsafe-action rate among automated decisions rises and then flattens, the flattening occurring because the highest-risk cases have by then already been absorbed into the automated arm. Panel (b) turns the trade-off into an expected cost per decision and shows the consequence that matters for design: the cost-optimal threshold is not fixed, it moves toward more human review as the ratio of harm cost to review cost grows, which is why a single default threshold across decision classes is rarely defensible. Panel (c) adds the capacity ceiling and shows how an escalation rate that looks prudent on paper can generate an unserved review queue once K is finite. The value of the figure is structural rather than numerical. The parameters were chosen to expose these relationships, not estimated from field data (Appendix B), and no value in it should be transferred to an operational threshold decision.

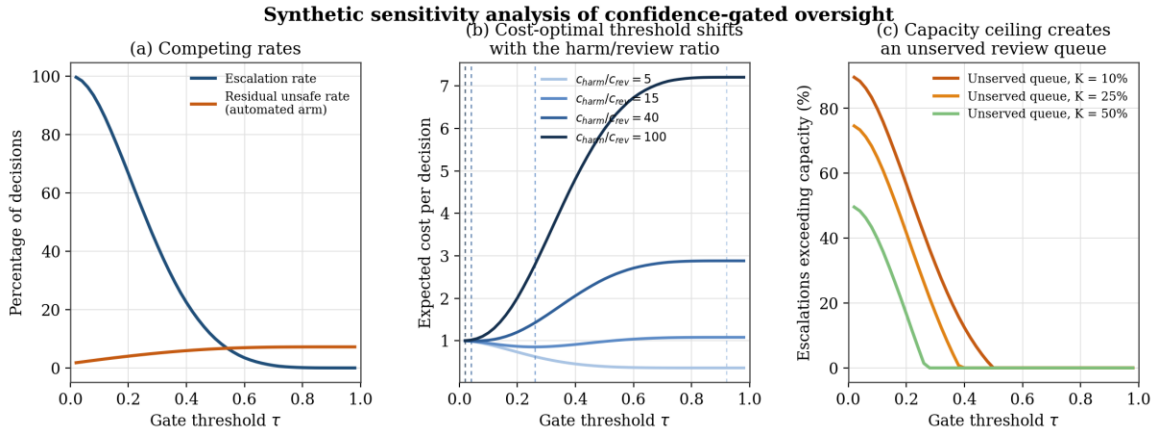


**Figure 3.** Synthetic sensitivity analysis of confidence-gated oversight. (a) Escalation rate and residual unsafe-action rate move in opposite directions as the gate threshold tau increases. (b) The cost-optimal threshold, marked by the dashed verticals, shifts toward more review as the harm-to-review cost ratio grows. (c) A finite reviewer capacity K leaves part of the escalation demand unserved at low thresholds. All three panels are model-based illustrations with fully specified parameters (Appendix B), not measured production results.

### 5.4 Traceability and runtime observability

Capture-at-commit and auditability [23] require more than log volume. Three properties are needed, and each fails in a characteristic way.

- **Trace completeness.** The fraction of committed actions for which the full decision-to-action-to-outcome chain resolves. A chain that resolves for 98 per cent of actions is not 98 per cent auditable, because the missing two per cent is not a random sample: it concentrates in retries, timeouts, and partial failures, which are the cases most likely to be under investigation.

- **Join integrity.** Identifiers must survive asynchronous and multi-step paths. This is the classical distributed-tracing problem addressed by infrastructure of the kind described in [26], with the added requirement here that the propagated identifier is the governed decision_id and not an arbitrary request identifier.
- **Retention alignment.** Traces must outlive the retention obligation attached to the decision. Otherwise the continuity constraint of Section 3.3 is satisfied at write time and violated at audit time, which is the only time it is tested.
  For agentic execution these properties must hold at trajectory level as well. A sequence of individually admissible steps can be jointly inadmissible, each below its own action threshold but cumulatively above it, which is the trajectory-risk concern Sazu raises [23] and a case the per-decision gate of Section 3.2 does not by itself cover.

## 6. Conformance Model

The unified architecture keeps the cumulative maturity structure of the Full-Stack architecture [23] and pairs each profile with the decision-assurance prerequisite that makes it meaningful. The profiles are cumulative: Governed presupposes Observable, and Adaptive presupposes Governed. A system should not claim adaptive learning if it cannot first demonstrate evidence binding, before-commit governance, and runtime traceability, because a learning loop built on an unjoinable evidence chain optimises against a signal it cannot attribute, and it will do so with full confidence.

| Profile | Decision-assurance prerequisite | Runtime capability | Representative exit criterion |
|---|---|---|---|
| Foundational / Observable | DecisionRecord persisted; source lineage registered | Capture-at-commit; trace completeness measured | Every execution joinable to its decision |
| Governed | Independent validation; calibrated confidence routing | Before-commit gating; compensation; readiness gates | All MUST controls satisfied and audited on a sample |
| Adaptive | Outcome-linked decision quality measured | Drift-under-action; retraining; off-policy evaluation | Loop improvement measurable without stability loss |

**Table 3.** Cumulative conformance profiles, adopted from [23] and paired with the decision-assurance prerequisite from [24] that each profile presupposes.

Figure 4 shows where each of Sazu's fifteen invariants is enforced across the seven lifecycle stages of the unified architecture. Reading the rows against the columns makes two things visible. The invariants are not redundant: with few exceptions each has a distinct primary enforcement point, and the supporting cells show where an invariant reaches beyond its home stage, as with evidence binding and auditability, both of which touch the observability spine. The two frameworks also interlock rather than overlap, with GDI concentrated on the left of the lifecycle and the Full-Stack invariants on the right, meeting at the pre-commit gate. That meeting point is the seam the shared evidence contract governs.

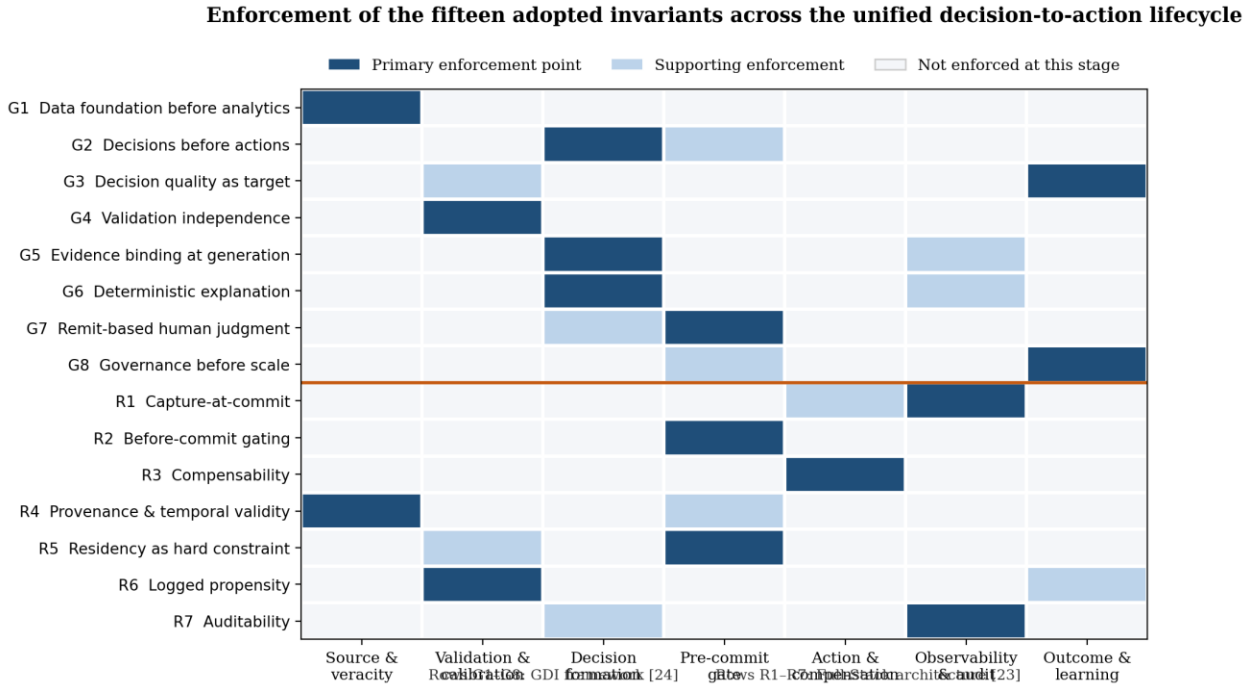


**Figure 4.** Enforcement of Sazu's fifteen adopted invariants (G1–G8 from GDI [24]; R1–R7 from the Full-Stack architecture [23]) across the seven stages of the unified lifecycle. A dark cell marks the primary enforcement point for an invariant; a light

cell marks a supporting one. The matrix records declared coverage and the division of labour between the two planes; it is not a performance score. The full mapping is given in Appendix C.

## 7. Evaluation Framework

Because the proposed architecture is a synthesis rather than a completed field study, and because the empirical programme the frameworks invite has not yet been run (Section 2.5), the evaluation design carries more weight here than usual. Evaluation proceeds at four levels, kept deliberately separate so that a gain at one level cannot be reported as evidence at another.

| Evaluation level | Metric family | Example measures | Research question |
|---|---|---|---|
| Conformance | Control validity | MUST-pass rate; recorded deviation count; evidence completeness | Does the system satisfy the architecture? |
| Operational | Reliability and traceability | p50/p95 latency; availability; trace completeness; input freshness | Can the system operate and reconstruct safely? |
| Decision | Quality and risk | Precision; calibration error; override rate; realised outcome quality | Does the governed system improve decisions? |
| Learning | Adaptation | Drift-detection latency; regression yield; off-policy value estimate | Does feedback improve the system without destabilising it? |

**Table 4.** Four-level evaluation framework. The conformance and operational levels test whether the architecture is present; the decision and learning levels test whether it is worth having.

A workable protocol compares an ungoverned or partially governed baseline against the unified architecture over the same decision population, with the outcome definition fixed in advance, the analysis preregistered, and the validation role held independent. The primary hypothesis is not that the unified architecture raises AUC, which it very likely will not; a governance layer that moved ranking quality would be doing something unexpected and worth investigating in its own right. The hypothesis is that it reduces latent governance failures: unreconstructable decisions, actions committed on expired inputs, explanations that do not match the executed rationale, and escalations absorbed by capacity limits. None of these appears in ordinary ranking metrics, which is the direct consequence of the separation between model performance and decision quality that Sazu makes an invariant [24].

Two measurement hazards deserve explicit treatment, because either one would make a governed system look better than it is.

- **Self-reported conformance.** Conformance rates are produced by the system under test. A MUST-pass rate near 100 per cent is as consistent with a permissive gate as with a well-governed one. Conformance therefore has to be audited against a sample of independently reconstructed decisions rather than accepted at face value, which is the internal-auditing discipline argued for in [20].
- **Population change induced by the intervention.** Escalation removes the hardest cases from the automated arm, so automated-arm outcome metrics improve mechanically even when the system as a whole has not. Comparisons must be made over the full decision population, including escalated and refused cases, with escalation volume reported next to every outcome figure.

## 8. Discussion

### 8.1 What the synthesis adds to the foundation

GDI states what must be true before a decision is trusted, and the Full-Stack architecture states what must be true while the platform acts on that decision [23], [24]. Neither framework, on its own, prevents the boundary between the two from becoming an accountability gap, the place where a validated model and an observable runtime coexist while no artifact connects the decision that was certified to the action that was actually taken. The contribution of this paper is narrow and sits exactly on that seam: the shared evidence contract of Section 4.3, the admissibility conjunction of Section 3.2, the placement of freshness evaluation at commit rather than at scoring, and the requirement that refusals be recorded as first-class events. The control structure underneath is Sazu's, adopted rather than reinvented, and the value of the synthesis depends on the quality of that foundation.

### 8.2 Relationship to established methods

The resulting architecture is a composition layer over established methods, not a replacement for any of them. Attribution stays with the explanation literature [21], [15], [4]; uncertainty with conformal and recalibration methods [27], [3], [13]; transparency with documentation conventions [16], [12]; traceability with distributed tracing [26]; compensation with the saga pattern [11]; and the taxonomy of failure with the production-ML and AI-safety literature [25], [1], [10], [2], [14]. The architectural claim is only that these become coordinated controls attached to a common decision and runtime evidence

chain, each with a defined boundary, a named evidence artifact, and a specified failure response. That is a claim about placement, and it should be assessed as one.

### 8.3 Why the decision-to-action boundary matters

A model can be perfectly reproducible and still produce an unsafe operational outcome: the wrong policy version is invoked, a stale source is read, the decision executes after a material context change, or the downstream action turns out not to be reversible. An exemplary observability platform, conversely, cannot establish that a decision was valid when the model-validation and evidence records do not exist. Sazu's two frameworks are complementary in a technically specific sense rather than being two descriptions of one system: they fail differently, and each covers the other's characteristic failure mode. That asymmetry, rather than any similarity between them, is what makes the composition worth making.

### 8.4 Adoption path

Implementation order matters, and it follows the conformance profiles of Section 6. Identifier propagation and capture-at-commit come first, because every later control is measured through them. Veracity gating and evidence binding follow, being inexpensive relative to their diagnostic value. Before-commit gating and compensation come next and are the most invasive, since they touch the action layer directly. Confidence-gated routing should not be switched on before calibration has been measured (Section 5.2) and reviewer capacity is known (Section 5.3). Adaptive learning comes last. Organisations that invert this order tend to deploy routing thresholds on uncalibrated scores and then cannot tell whether the resulting escalation volume reflects genuine risk or miscalibration, a question that becomes unanswerable once the thresholds have been tuned against the confounded signal.

## 9. Threats to Validity and Limitations

- **No field evaluation.** The unified architecture has not been evaluated in a live production environment. The synthetic sensitivity analysis is illustrative and must not be cited as field evidence.
- **Foundation not yet field-tested.** As Section 2.5 sets out, Sazu's frameworks are normative and their field benefit has not yet been measured under controlled conditions. Any benefit claimed for this synthesis inherits that evidential status.
- **Inherited method assumptions.** Conformal guarantees hold only under their stated exchangeability assumptions [27], [3]; calibration is relative to the evaluation population; drift thresholds are domain-specific; and reviewer capacity can become the binding constraint on the entire oversight design.
- **Construct validity.** Conformance measures declared compliance, not safety. Section 4.4 lists failure modes that occur entirely inside formally satisfied controls and are invisible to conformance evidence.
- **Censored learning signal.** Outcome ownership frequently sits outside the platform, truncating the learning loop. Worse, outcomes are usually observed only for decisions that were acted upon, so the loop is trained on a selectively censored sample and will systematically underestimate the cost of the actions it did not take.
- **Limited generality.** The formulation assumes a discrete, named decision with a bounded action set. Continuous control, ranking and recommendation surfaces with diffuse action boundaries, and open-ended generative systems fit less cleanly, and extending the admissibility predicate to them is open work.

These limitations describe a concrete empirical programme: preregister greenfield implementations, measure identical decision populations before and after the governance controls, keep an independent validation role, report conformance, operational, decision, and learning outcomes separately, and publish escalation volumes next to every outcome figure so that the population-change hazard of Section 7 can be assessed by readers rather than assumed away by authors.

## 10. Conclusion

This paper set out a unified decision-to-action governance architecture built squarely on the foundation Sazu established in 2023: the Governed Decision-Intelligence framework [24] and the Full-Stack Production-Platform Reference Architecture [23]. GDI supplies the decision-assurance plane, with veracity, independent validation, decision quality as the assurance target, generation-time evidence, deterministic explanation where required, and confidence-gated oversight. The Full-Stack architecture supplies the execution-assurance plane, with capture-at-commit, before-commit gating, compensability, provenance and temporal validity, residency, logged propensity, auditability, trajectory risk, drift-under-action, and operational readiness. The strength of the design is inherited from the strength of those two frameworks, and adopting them was a deliberate choice rather than a default.

The synthesis contributed here is the interface between the two planes: a shared evidence contract, an admissibility conjunction evaluated strictly before external commitment, freshness evaluated at commit rather than at scoring, and refusals recorded as first-class events. The resulting design treats the decision-to-action path as one governed lifecycle in which a DecisionRecord establishes why a decision was trusted, an ExecutionRecord establishes how it was carried out, and an outcome join establishes what followed. The practical consequence is that governance becomes testable not only at model promotion but at every consequential boundary where an AI system changes the state of the world.

The architecture is not yet field-tested, and neither is the foundation it adopts. The most useful next step is not further architectural elaboration but evaluation: validating the design across regulated decision services, enterprise automation, and agentic systems, quantifying its effect on audit effort and on decision quality separately, and establishing how adaptive learning can be enabled safely once the evidence and execution planes are demonstrably mature. Until that work exists, the contribution here should be read as a falsifiable proposal that rests on a strong foundation, rather than as a demonstrated result.

**Declarations**

**Funding.** No funding was received for this work.

**Competing interests.** The author declares no competing interests. This paper depends substantially on the two frameworks published by M. H. Sazu [23], [24], which it adopts as its foundation. That dependence is stated openly in Section 1.3, and the current limits of the evidence base are set out in Section 2.5.

**Data availability.** No empirical data were used. The synthetic scenario behind Figure 3 is fully specified in Appendix B and is reproducible from that specification alone.

**Author contributions.** Conceptualisation, formal analysis, and writing were carried out by the author. The GDI and Full-Stack frameworks are the prior work of M. H. Sazu and are adopted here under attribution.

**References**

*References are listed alphabetically by first-author surname; in-text citation numbers follow this ordering.*

**[1]** Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019). Software Engineering for Machine Learning: A Case Study. Proceedings of the 41st International Conference on Software Engineering (ICSE-SEIP), 291–300. DOI: 10.1109/ICSE-SEIP.2019.00042.

**[2]** Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete Problems in AI Safety. arXiv:1606.06565.

**[3]** Angelopoulos, A. N., & Bates, S. (2021). A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification. arXiv:2107.07511.

**[4]** Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. Information Fusion, 58, 82–115. DOI: 10.1016/j.inffus.2019.12.012.

**[5]** Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (2016). Site Reliability Engineering: How Google Runs Production Systems. O'Reilly Media.

**[6]** Board of Governors of the Federal Reserve System & Office of the Comptroller of the Currency. (2011). Supervisory Guidance on Model Risk Management (SR 11-7 / OCC Bulletin 2011-12).

**[7]** Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction. Proceedings of the IEEE International Conference on Big Data, 1123–1132. DOI: 10.1109/BigData.2017.8258038.

**[8]** Doshi-Velez, F., & Kim, B. (2017). Towards a Rigorous Science of Interpretable Machine Learning. arXiv:1702.08608.

**[9]** European Parliament and Council. (2024). Regulation (EU) 2024/1689 Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act). Official Journal of the European Union.

**[10]** Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A Survey on Concept Drift Adaptation. ACM Computing Surveys, 46(4), 1–37. DOI: 10.1145/2523813.

**[11]** Garcia-Molina, H., & Salem, K. (1987). Sagas. ACM SIGMOD Record, 16(3), 249–259. DOI: 10.1145/38714.38742.

**[12]** Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2021). Datasheets for Datasets. Communications of the ACM, 64(12), 86–92. DOI: 10.1145/3458723.

**[13]** Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On Calibration of Modern Neural Networks. Proceedings of the 34th International Conference on Machine Learning (ICML), 70, 1321–1330.

**[14]** Hendrycks, D., Carlini, N., Schulman, J., & Steinhardt, J. (2022). Unsolved Problems in ML Safety. arXiv:2109.13916.

**[15]** Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems (NeurIPS), 30, 4765–4774.

**[16]** Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model Cards for Model Reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT*), 220–229. DOI: 10.1145/3287560.3287596.

**[17]** Naeini, M. P., Cooper, G. F., & Hauskrecht, M. (2015). Obtaining Well Calibrated Probabilities Using Bayesian Binning. Proceedings of the 29th AAAI Conference on Artificial Intelligence, 2901–2907.

**[18]** National Institute of Standards and Technology. (2023). Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1. DOI: 10.6028/NIST.AI.100-1.

**[19]** Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2022). Challenges in Deploying Machine Learning: A Survey of Case Studies. ACM Computing Surveys, 55(6), 1–29. DOI: 10.1145/3533378.

**[20]** Raji, I. D., Smart, A., White, R. N., Mitchell, M., Gebru, T., Hutchinson, B., Smith-Loud, J., Theron, D., & Barnes, P. (2020). Closing the AI Accountability Gap: Defining an End-to-End Framework for Internal Algorithmic Auditing. Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT*), 33–44. DOI: 10.1145/3351095.3372873.

**[21]** Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135–1144. DOI: 10.1145/2939672.2939778.

**[22**] Rudin, C. (2019). Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. Nature Machine Intelligence, 1, 206–215. DOI: 10.1038/s42256-019-0048-x.

**[23]** Sazu, M. H. (2023). A Full-Stack Production-Platform Reference Architecture For Governed Agentic And Automated-Action Enterprise Systems. IPHO-Journal Of Advance Research In Science And Engineering, 1(12), 49-58.

**[24]** Sazu, M. H. (2023). Governed Decision-Intelligence (GDI). Ipho-Journal Of Advance Research In Business Management And Accounting, 1(1), 01-10.

**[25]** Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden Technical Debt in Machine Learning Systems. Advances in Neural Information Processing Systems (NeurIPS), 28, 2503–2511.

**[26]** Sigelman, B. H., Barroso, L. A., Burrows, M., Stephenson, P., Plakal, M., Beaver, D., Jaspan, S., & Shanbhag, C. (2010). Dapper, a Large-Scale Distributed Systems Tracing Infrastructure. Google Technical Report dapper-2010-1.

**[27]** Vovk, V., Gammerman, A., & Shafer, G. (2005). Algorithmic Learning in a Random World. Springer. DOI: 10.1007/b106715.

## Appendix A. Reference Implementation Checklist

Each check names the adopted invariant it operationalises, so that an implementer can trace any control back to its source framework. G-numbers refer to the GDI invariants [24]; R-numbers to the runtime invariants [23]; entries marked "this paper" are the extensions introduced in Sections 3 and 4.

| ID | Implementation check | Pass condition | Basis |
|---|---|---|---|
| A1 | Source registry | Every production source carries owner, provenance, freshness, and validity metadata | G1 |
| A2 | Veracity gate | Sub-threshold inputs are blocked or quarantined before model scoring, and q_min is recorded with the verdict | G1, R4 |
| A3 | Model validation | Promotion requires an independent, preregistered protocol with stated blocking criteria | G4 |
| A4 | Calibration | Confidence used for routing is empirically calibrated for the intended population and reported by subgroup | G4 |
| A5 | DecisionRecord | Every acted-upon decision has generation-time evidence and complete version references | G5 |
| A6 | Explanation | Regulated explanations reproduce the recorded rationale deterministically from the persisted record | G6 |
| A7 | Before-commit gate | The full admissibility conjunction A(d) is evaluated before any external side effect | G2, R2 |
| A8 | ExecutionRecord | Runtime execution is joinable to its decision record by shared identifier, not by inference | R1, R7 |
| A9 | Compensation | External effects are idempotent, transactional, or compensable, with the compensation reference recorded | R3 |
| A10 | Recorded refusal | Blocked commits record the failing conjunct, its evidence, and the response taken | This paper (Section 4.3) |
| A11 | Outcome loop | Realised outcomes are joined to decisions and can trigger drift or retraining controls | G8, R6 |
| A12 | Trajectory admissibility | Multi-step agentic paths are assessed for cumulative risk, not only per-step risk | R2 (trajectory extension) |
| A13 | Reviewer capacity | Escalation volume implied by tau is within reviewer throughput K, and breaches are alerted | G7, R2 |
| A14 | Readiness | SLOs, error budgets, runbooks, and recovery controls are assessed before general availability | R7 |
| A15 | Retention alignment | Traces and version references outlive the retention obligation of the decision class | R7 (Section 5.4) |

**Table A1.** Reference implementation checklist with basis attribution to Sazu's adopted invariant sets.

## Appendix B. Reproducibility Note for the Synthetic Sensitivity Analysis

Figure 3 is generated from a fixed-seed Monte Carlo scenario over 200,000 synthetic decisions. A latent risk score r is drawn from Beta(2.2, 5.5). The probability that an automated action is unsafe is parameterised as 0.015 + 0.20r, so that risk rises linearly in the latent score from a non-zero floor. For a given threshold tau, cases with r > tau are escalated to human review and cases with r <= tau are automated. Panel (a) plots the escalation fraction over the whole population and the residual unsafe-action fraction among automated decisions. Panel (b) plots the expected cost per decision, c_harm times the residual unsafe volume plus c_rev times the escalation fraction, for harm-to-review cost ratios of 5, 15, 40, and 100, with the cost-minimising threshold marked. Panel (c) plots, for reviewer capacities K of 10, 25, and 50 per cent of the population, the escalation demand that exceeds capacity.

Three properties of the specification should be understood before the figure is read. The Beta(2.2, 5.5) prior is right-skewed, which is why the escalation curve falls steeply at low tau. The linear risk function imposes a monotone relationship between the latent score and harm that real systems need not exhibit. And in panels (a) and (b) human review is modelled as perfectly

effective and of unbounded capacity, which is the very assumption panel (c) and Section 5.3 argue against; relaxing it flattens any apparent benefit from lowering tau.

*These parameters are illustrative and were chosen to expose the governance trade-off. They are not estimated from field data, and no numerical value in Figure 3 should be transferred to an operational threshold decision.*

## Appendix C. Mapping of Adopted Invariants to the Unified Lifecycle

The unified architecture carries fifteen normative invariants: eight from GDI [24] and seven from the Full-Stack architecture [23]. Table C1 records where each is enforced in the architecture of Sections 3 to 6 and, where applicable, which conjunct of the admissibility predicate A(d) expresses it. This mapping is the basis for Figure 4, and it records declared coverage and the division of labour between the two planes rather than assurance strength.

| ID | Invariant as stated by Sazu | Enforcement point in the unified architecture | Formal locus |
|---|---|---|---|
| G1 | Data foundation before analytics | Veracity gate on entry to the model path (Section 5.1) | V(x) |
| G2 | Decisions before actions | Named decision formed before action selection (Section 3.2) | Ev(d) |
| G3 | Decision quality as the assurance target | Decision-level evaluation kept separate from model metrics (Sections 2.3, 7) | Evaluation level 3 |
| G4 | Validation independence | Independent preregistered protocol gates promotion (Section 5.2) | Val(m) |
| G5 | Evidence binding at generation time | DecisionRecord persisted before hand-off (Section 3.3) | Ev(d) |
| G6 | Deterministic explanation where regulated | Explanation boundary in the decision plane (Section 4.1) | Ev(d), explanation_ref |
| G7 | Remit-based human judgment | Routing and reviewer remit under capacity K (Section 5.3) | H(d) |
| G8 | Governance before scale, with outcome loop | Cumulative conformance profiles (Section 6) | Profile gating |
| R1 | Capture-at-commit | ExecutionRecord written at commit_ts (Section 5.4) | E_X(d) |
| R2 | Before-commit gating | Full conjunction evaluated pre-commit (Section 3.2) | A(d) |
| R3 | Compensability | Idempotency, transaction, or compensation required (Section 4.2) | Comp(a) |
| R4 | Provenance and temporal validity | Freshness re-checked at commit, not at scoring (Section 3.2) | Fresh(x,t) |
| R5 | Residency as a hard constraint | Gateway residency filter within policy resolution (Section 4.2) | Pol(p,a) |
| R6 | Logged propensity | Routing choices logged for off-policy evaluation (Sections 4.2, 7) | Evaluation level 4 |
| R7 | Auditability | Evidence continuity constraint and retention alignment (Sections 3.3, 5.4) | id(E_D) = id(E_X) |

**Table C1.** Sazu's fifteen adopted invariants and their enforcement points in the unified decision-to-action architecture.