

Proportionate Governance in Practice

An IDEA–CRUS-T Framework for Risk-Tiered, Runtime-Governed AI Systems in Financial Services

Satyaki Ghosh Dastidar

AI Risk Consulting · satyaki.ghoshdastidar@airisk-consulting.com

Preprint. Under review. Comments welcome.

Abstract

Financial institutions increasingly deploy generative and agentic AI across use cases that range from internal drafting assistance to autonomous transaction execution. Applying uniform governance controls across this range produces two simultaneous failure modes: high-reach, low-authority systems are over-governed into shadow deployment, while high-authority systems inherit lightweight controls unsuited to their consequence. We argue that proportionate governance is a design property of the control architecture rather than a discretionary adjustment applied at review time, and we operationalize this claim through two linked instruments. The first, IDEA (Identification, Development/Deployment, Explainability/Execution, Accountability), is a lifecycle spine common to every AI system regardless of risk. The second is a five-dimension, capability-level risk-tiering method (Authority, Impact, Agency, Exposure, Recoverability) that assigns a discrete control tier at design time, and CRUS-T, a normalized, programmatically computable runtime reliability score across four dimensions — Consistency, Robustness, Uncertainty awareness, and Safety — gated against tier-specific thresholds and evaluated on a conservative statistical bound rather than a point estimate. Each dimension decomposes into standard, auditable sub-metrics (self-consistency agreement, performance-retention ratios, calibration error, selective-prediction quality, claim-level groundedness, and policy-conformance rate) aggregated by a collapse-sensitive geometric mean, with an asymmetric hard gate on critical safety violations. We illustrate the framework with two contrasting, hypothetical-but-representative case studies drawn from retail banking and anti-money-laundering operations, showing how capability-level CRUS-T scoring identifies the specific reliability dimension responsible for under-governance in one case and the single authority-bearing action responsible for over-governance in the other. We situate the contribution relative to existing model risk management guidance, agentic-risk taxonomies, runtime governance architectures, and agent reliability science, and discuss limitations — principally evaluation-set representativeness, parameter governance, and the dependency of learned verification components on their own validation — together with directions for future work.

Keywords: model risk management; agentic AI; runtime governance; risk tiering; reliability evaluation; calibration; selective prediction; retrieval-augmented generation; financial services regulation.

Contents

1	Introduction	4
1.1	Motivation	4
1.2	Contributions	4
1.3	Paper organization	5

2	Related Work	5
2.1	Model risk management baselines	5
2.2	Agentic governance and capability-centric frameworks	6
2.3	Runtime governance and integrated oversight	6
2.4	Agent reliability and evaluation practice	7
2.5	Agent security	7
2.6	Positioning of this work	7
3	The IDEA Governance Lifecycle	7
3.1	The four stages	8
3.2	Two governing instruments	9
4	Risk Tiering: From Five Dimensions to Four Control Tiers	10
4.1	The five dimensions	10
4.2	Four control tiers	11
4.3	The aggregation matrix: from five scores to a tier	11
4.4	Incremental residual risk and inherit-by-default	13
5	The CRUS-T Reliability Score	13
5.1	Design principles and normalization primitives	14
5.2	Consistency (C)	14
5.3	Robustness (R)	15
5.4	Uncertainty awareness (U)	15
5.5	Safety (S) and the critical-violation gate	17
5.6	Composite score, the tier gate, and confidence-bound gating	18
5.7	Runtime zones	18
5.8	Deployment architecture: synchronous, asynchronous, and batch evaluation	20
6	Case Studies	20
6.1	Case 1: an under-governed non-agentic system	21
6.2	Case 2: an over-governed agentic system	22
6.3	Cross-case synthesis	24
7	Discussion	24
7.1	Proportionality as a control-plane property, not a review outcome	24
7.2	Operating-model implications	24
7.3	Governance as an accelerant, not a tax	25
7.4	MRM as standard-setter and challenger	25
8	Limitations	25
9	Future Work	26
10	Conclusion	27
A	Notation Summary	30
B	Reference Computation (Pseudocode)	30
C	Extended Worked Example	31

D Extended Case-Study Tables

32

1 Introduction

1.1 Motivation

Model risk management (MRM) programs at large financial institutions were built for a world of stable input–output mappings: a model consumes features and produces a score, and governance validates that mapping prior to deployment and monitors its stability thereafter. Generative and agentic AI systems break this assumption in two ways. First, outcomes arise from an execution trajectory — a sequence of reasoning steps, tool invocations, and state transitions — rather than a single evaluable mapping, so severe failures can occur on specific trajectories even when aggregate performance metrics look acceptable [1, 2]. Second, the natural response to this shift — applying uniform, conservative controls to every AI use case — produces two failure modes simultaneously rather than a single one. Low-authority, high-volume systems (an internal summarization tool, a document-drafting assistant) are queued behind review processes calibrated for regulatory capital models, creating multi-month approval cycles for multi-week engineering efforts. Faced with that friction, business units route around governance, producing exactly the unmonitored “shadow AI” deployment the process was meant to prevent. Simultaneously, high-authority, high-consequence systems (autonomous transaction agents, customer-facing decisioning tools) inherit governance calibrated for the low-risk majority of use cases, because the intake process asked the wrong question — typically some variant of “is this agentic?” — rather than “what is the consequence of failure, and can it be contained?”

The standard prescription for this tension is “risk-based” or “proportionate” governance: calibrate control intensity to risk. This prescription is correct but, as typically implemented, incomplete. Proportionality is usually applied as a discretionary adjustment negotiated between the first and second lines of defense (1LoD, 2LoD) on a case-by-case basis — a committee decides that a given use case merits a lighter review. This negotiation does not scale: it consumes second-line capacity linearly in the number of use cases, produces inconsistent outcomes across similar systems reviewed by different committees or at different times, and leaves no artefact by which a regulator or internal auditor can verify that the tiering decision was principled rather than expedient.

This paper takes the position that proportionality must be a **design property of the governance architecture** — computed from measurable properties of a use case and its constituent capabilities, applied consistently through a small number of discrete control tiers, and re-verified continuously at runtime rather than assessed once and assumed to hold. We operationalize this position through two linked instruments that this paper develops in full: a five-dimension, capability-level risk-tiering method that fixes the control tier at design time, and a four-dimension, normalized runtime reliability score — CRUS-T — that verifies the tier’s assumptions continue to hold in production and triggers escalation or containment when they do not.

1.2 Contributions

This paper makes the following contributions:

- A lifecycle frame, **IDEA** (Identification, Development/Deployment, Explainability/Execution, Accountability), that is common to every AI system irrespective of risk, separating the *where* of governance (which stage of the lifecycle a control belongs to) from the *how much* (the depth of control at that stage), which is set by the two instruments below.
- An operationalization of a five-dimension risk-tiering method — Authority, Impact, Agency, Exposure, Recoverability — building on and extending the capability-centric governance view of Szpruch et al. [1] and the risk-and-capability taxonomy of Khoo et al. [3], into four discrete control tiers with explicit, auditable control substitutions at each tier.

- A fully specified, **normalized and programmatically derivable** construction of the four reliability dimensions identified qualitatively by Rabanser et al. [4] — Consistency, Robustness, Uncertainty awareness, and Safety — each reduced to standard, auditable sub-metrics (self-consistency agreement, retention ratios, calibration error, selective-prediction quality, claim-level groundedness, policy-conformance rate), aggregated by a collapse-sensitive weighted geometric mean, and gated on a bootstrap lower confidence bound rather than a point estimate.
- An explicit, asymmetric treatment of safety: a discontinuous hard gate that zeroes the governed safety score on any critical policy violation, formalizing the principle that some failure modes are not survivable in expectation and must not be averaged away by an otherwise strong composite score.
- Two contrasting, hypothetical-but-representative case studies — a retail-banking conversational assistant and an anti-money-laundering (AML) alert-triage agent — demonstrating how capability-level CRUS-T scoring pinpoints *which* reliability dimension is responsible for under-governance in one case, and *which single authority-bearing action* is responsible for over-governance in the other, in each instance replacing a label-driven tiering decision (“non-agentic,” “agentic”) with a measurement-driven one.

1.3 Paper organization

Section 2 reviews related work across model risk management baselines, agentic governance and capability-centric frameworks, runtime governance architectures, agent reliability and evaluation science, and agent security. Section 3 develops the IDEA lifecycle. Section 4 develops the five-dimension risk-tiering method and its four control tiers. Section 5 is the technical core of the paper: the full mathematical construction of the CRUS-T reliability score. Section 6 presents the two case studies. Sections 7 to 9 discuss implications, limitations, and future work, and Section 10 concludes. Appendices provide notation, reference pseudocode, and extended numerical detail.

2 Related Work

This work sits at the intersection of five overlapping literatures: model risk management baselines, agentic governance and capability-centric frameworks, runtime governance and integrated oversight, agent reliability and evaluation science, and agent security. We review each in turn and position our contribution against them.

2.1 Model risk management baselines

Supervisory guidance on model risk management — SR 26-2, which superseded SR 11-7 in 2026, and its companion OCC guidance in the United States [5, 6], PRA SS1/23 in the United Kingdom [7], and OSFI Guideline E-23 in Canada [8] — establishes the baseline expectations this paper takes as given: model inventory, documented intended use and limitations, independent validation evidence, ongoing monitoring, and governed change control. These frameworks were written for a broad class of statistical and machine-learning models and are technology-agnostic by design; their requirements remain binding for agentic systems, but they specify *what* must be governed without specifying *how* to operationalize governance for systems whose behavior is defined by an execution trajectory rather than a static mapping. General AI risk frameworks — NIST’s AI RMF [9] and the EU AI Act [10] — provide complementary lifecycle and risk-categorization obligations but are similarly not agentic-specific in their technical governance primitives.

At the international coordination level, the Financial Stability Board (FSB) has assessed AI’s implications for financial stability since 2017 [11], updating its stocktake in November 2024 [12] and, in October

2025, publishing guidance on monitoring AI adoption and associated vulnerabilities, including concentration risk arising from financial institutions’ reliance on a small number of third-party model and infrastructure providers [13]. Most directly relevant to this paper, the FSB’s June 2026 consultation report proposes a menu of twelve sound practices for responsible AI adoption, organized into organization-wide governance (practices 1–4), risk management across the stages of AI development and deployment (practices 5–10), and AI-related cyber, ICT, and third-party risk (practices 11–12), and explicitly invites comment on whether the practices strike the right balance for “emerging and new complex forms of AI, such as GenAI and agentic AI” [14]. Notably, the FSB frames its own included case studies as illustrations of how the sound practices can be applied *proportionately* — the same objective this paper takes as its starting point, at the level of an individual institution’s control architecture rather than a supervisory toolkit. As of this writing the report remains open for comment (responses due 22 July 2026), with a final version expected in October 2026 as a US G20 deliverable; consistent with its own description as non-binding and non-prescriptive, it specifies governance objectives and lifecycle stages without prescribing a computable scoring or tiering mechanism to operationalize them across those stages. The IDEA lifecycle (Section 3) and CRUS-T score (Section 5) map directly onto this stage-based governance structure and are designed to supply exactly this kind of per-stage, auditable evidence.

This paper is best read as an operationalization layer beneath these frameworks and the FSB’s sound practices: the tiering method and CRUS-T score are designed to produce the auditable evidence they require, for the specific case of agentic and generative AI.

2.2 Agentic governance and capability-centric frameworks

Two recent works anchor the capability-centric view this paper builds on. Khoo et al. [3] introduce the Agentic Risk & Capability (ARC) framework, arguing that the primary risk surface of an agentic system is its capabilities and orchestration design rather than the base model alone, and motivate tiered controls based on qualitative impact/likelihood scoring. Szpruch et al. [1] extend this into a full bank MRM operating model: capabilities as the unit of validation and reuse, pooled evidence across workflows, a five-dimension risk-tiering method (Authority, Impact, Agency, Exposure, Recoverability), and a shift from periodic model validation to continuous runtime governance over execution trajectories, formalized as a labeled transition system with deterministic guard conditions. This paper adopts that five-dimension tiering method directly (Section 4) and adopts the qualitative reliability dimensions — Consistency, Robustness, Uncertainty awareness, Safety — that Rabanser et al. [4] identify as necessary to avoid average task-success metrics masking operationally important failure modes. Our contribution relative to both is the **full mathematical specification** of those reliability dimensions as normalized, programmatically computable sub-metrics with an explicit aggregation rule, a governance-appropriate statistical treatment (bootstrap lower-bound gating), and worked demonstrations of how the resulting score changes a tiering decision in practice.

2.3 Runtime governance and integrated oversight

MI9 [15] provides a unified runtime governance architecture for agentic AI written from a bank MRM perspective, integrating governance-semantic telemetry, authorization monitoring, conformance checking, drift detection, and containment — the operational substrate that a runtime score such as CRUS-T is designed to consume and act upon. This paper treats that telemetry layer as given (OpenTelemetry-style traces and spans [16]) and focuses on the scoring and gating logic that sits atop it: what specifically gets measured, how it is normalized and combined, and at what threshold a trajectory triggers escalation or containment.

2.4 Agent reliability and evaluation practice

Beyond the qualitative four-dimension framing of Rabanser et al. [4], our sub-metric constructions draw on established, independently validated evaluation constructs. For groundedness and faithfulness we adopt the claim-level entailment construction popularized by RAGAS [17]. For calibration we use expected calibration error following Guo et al. [18], whose central finding — that modern high-capacity models are frequently overconfident relative to their accuracy — motivates treating calibration as a first-class, separately monitored dimension rather than an afterthought. For selective prediction we adopt the risk–coverage / area-under-risk-coverage-curve (AURC) construction of Geifman and El-Yaniv [19]. For self-consistency we draw on the sampling-agreement construction of Wang et al. [20], originally proposed to improve chain-of-thought reasoning and repurposed here as a diagnostic of output stability. For categorical agreement we use Krippendorff’s α [21], the standard chance-corrected inter-rater statistic, applied to a model as its own repeated rater across samples. Failure-mode-driven evaluation as a compounding lifecycle asset — incidents promoted into regression tests and monitoring signals — follows the practice described by Anthropic [22] and by Sudjianto and Wingyan [23] in the context of structured financial-document evaluation.

2.5 Agent security

Work on state-dependent agent vulnerabilities [24] and tool-use safety [25, 26, 27, 28, 29, 30] establishes that agentic systems are vulnerable at specific execution states and that backbone vulnerabilities can be amplified through tool use and memory, motivating the adversarial-suite construction of the Robustness dimension (Section 5.3) and the authorization-audit construction of the Safety dimension (Section 5.5). We treat this literature as the source of the adversarial test suites a Robustness score should be evaluated against, rather than re-deriving attack taxonomies here.

2.6 Positioning of this work

Relative to this literature, the contribution of this paper is narrow and specific: it does not propose a new agentic architecture, a new class of attacks, or a new regulatory taxonomy. It proposes a fully specified, normalized, and auditable **measurement instrument** — CRUS-T — for the reliability dimensions that the literature has identified as necessary but has not, to our knowledge, reduced to a common, governance-ready construction with an explicit aggregation rule, an asymmetric safety treatment, and a statistically conservative gating criterion suitable for a regulated decision. We view this as complementary to, and dependent upon, the capability-centric tiering and runtime-governance architectures of Szpruch et al. [1] and Wang et al. [15]: those works define *what* to govern and *where* enforcement occurs; this paper defines *how to measure* whether governance is holding.

3 The IDEA Governance Lifecycle

Every AI system deployed by a financial institution — from a low-authority internal drafting assistant to a high-authority autonomous transaction agent — passes through the same four lifecycle stages. We label these Identification, Development/Deployment, Explainability/Execution, and Accountability (IDEA). The purpose of naming this spine explicitly is to separate two questions that are routinely conflated in governance practice: *where* in the lifecycle a control belongs, which does not vary with risk, and *how much* control is applied at that point, which does. Sections 4 and 5 supply the two instruments — a design-time risk tier and a runtime reliability score — that answer the second question; this section defines the stages the instruments attach to.

3.1 The four stages

- **Identification.** What is the system, what is it authorized to touch, and what can it decide or do without further human input? This stage produces the capability inventory and authority scope that feed the risk-tiering method of Section 4.
- **Development / Deployment.** How is the system built, tested, and brought to production — and what evidence must exist at each step before the next is permitted? The stage divides into two sub-stages whose content varies by provenance but whose governance purpose is identical: ensure that the technical assumptions embedded in the system are documented, tested, and sufficient to support the assigned control tier before any production authority is granted.

The *Development* sub-stage applies to internally developed or materially modified systems. It covers design and architecture choices (prompt engineering, retrieval design, tool integration, orchestration logic), training or fine-tuning decisions where the institution modifies a base model, evaluation against held-out benchmarks representative of the production distribution, and pre-production independent validation proportionate to the assigned tier. The depth of each activity is tier-conditioned: a T1 internal assistant requires lightweight fit-for-purpose testing; a T3 compliance-adjacent capability requires formal validation evidence, documented intended use and limitations, and independent challenge of the test methodology; a T4 autonomous system requires validation rigor equivalent to a regulatory capital model, including out-of-distribution stress testing and formal sign-off from the second line before any production authority is activated. For off-the-shelf or minimally modified vendor solutions, the *Development* sub-stage is substantially abbreviated or replaced by vendor due diligence: contractual evidence of the vendor’s own validation practices, third-party audit reports where available, and institution-run fit-for-purpose testing on production-representative data. Critically, vendor provenance does not eliminate *Development*-stage governance obligations; it transfers part of the evidence burden to third-party assurance and shifts the institution’s own testing toward integration behavior and deployment-context fitness rather than intrinsic model properties.

The *Deployment* sub-stage is universal: it is the design-time control gate that every system must clear before reaching production, regardless of provenance. It encompasses the formal approval decision (sign-off authority conditioned on the tier), the production-ready control package specified by the tier, and the establishment of the monitoring and escalation infrastructure that the Explainability/Execution and Accountability stages will operate against. Deployment is where the evidence assembled during *Development* is reviewed, accepted or challenged, and converted into a conditional production authorization.

- **Explainability / Execution.** Can the institution see how the system reasons, and does the system operate inside enforced guardrails while it acts? The paired label is deliberate: for agentic systems, runtime execution control is at least as important as post-hoc explainability, and the two obligations are complementary rather than substitutable. This stage is scoped strictly to *per-trajectory*, *inference-time* concerns: the telemetry emitted by a single execution, the guardrails evaluated during that execution, and the CRUS-T score computed over that trajectory’s outputs.

The *Explainability* obligation asks whether a human reviewer — a compliance officer, a second-line validator, a regulator, or a customer-facing disputes team — can reconstruct what happened on a specific trajectory after the fact. For a single-step generative system, this means capturing the input, the retrieved context (where applicable), the output, and the grounding source that the output stood on. For a multi-step agentic system, it means capturing the full reasoning trace: the sequence of intermediate reasoning steps or chain-of-thought, each tool invocation with its inputs and outputs, the state visible to the model at each decision point, and the final action or output together with the evidence chain that produced it.

The depth of this trace requirement is tier-conditioned: a T1 assistive tool requires output logging and factual-error review; a T3 governed capability requires a full, reproducible reasoning trace that identifies the source document or data record behind each claim in the output; a T4 critical-autonomous system requires a tamper-evident audit log whose integrity can be verified independently of the system that produced it. Explainability in this sense is not an interpretability technique applied to model internals — it is an *operational governance record* of what the system did and why, scoped to the trajectory level, sufficient to support regulatory examination, complaint resolution, and incident investigation without requiring access to the model weights or training data.

The *Execution* obligation asks whether the system operates inside its authorized envelope at the moment of action, not merely whether it passed a pre-production gate. This is the enforcement complement to the approval authority established at Deployment: while the Deployment gate determines what the system is permitted to do in principle, the Execution controls enforce that permission boundary on every trajectory in production. They include synchronous policy-as-code guards that block unauthorized tool calls or output patterns before they are released (e.g. fee-rate claims not verified against a source-of-truth register, dispatch actions not preceded by a logged approval event), tool allowlists that restrict what the system can invoke at runtime, authority-scope enforcement that prevents the system from acting on data it was not authorized to access, and the CRUS-T reliability score of Section 5 that computes — per trajectory — whether the system’s outputs remain within the reliability envelope the tier requires. When a synchronous guard fires or the CRUS-T score breaches a threshold, the response is immediate and tier-conditioned: escalation (human review triggered by the score, preserving a partial, informative trace) or containment (restricting the system to a safe operating mode before the next trajectory is permitted). The critical design principle is that execution controls must be computable functions of governed state — deterministic guards over logged authorization events, tool-call records, and output content — and must not rely on a probabilistic model to verify another probabilistic model, since such a verifier is structurally blind to the fluent-but-wrong outputs it is meant to detect.

Together, Explainability and Execution close the same governance gap from opposite sides: Explainability ensures the institution can account for what the system did; Execution ensures the system does not exceed what it was authorized to do. Neither is sufficient alone. A system with full traceability but no runtime enforcement can produce a perfect audit trail of a policy violation. A system with strong enforcement but no trace cannot be examined after a failure, cannot support regulatory inquiry, and cannot route incidents to the correct remediation path.

- **Accountability.** Who owns the outcomes, monitors for drift, and is answerable when the system fails? This stage is scoped strictly to *population-level, longitudinal* concerns: monitoring across trajectories over time, drift detection, model-inventory management, escalation routing, human-in-the-loop design, and the named-role ownership that makes containment decisions when the Explainability/Execution stage surfaces an anomaly. The boundary between the two stages is temporal: E asks “is this trajectory safe right now?”; A asks “is the population of trajectories still behaving as expected, and who acts when it is not?”

3.2 Two governing instruments

The lifecycle stage a control belongs to does not change with the risk of the system; the depth of control applied at that stage does. We separate that depth into a design-time decision and a runtime decision, illustrated in Figure 1. At *Identification*, the five-dimension risk-tiering method of Section 4 assigns a discrete control tier. That tier then governs both sub-stages of *Development/Deployment*: it determines the depth of pre-production validation and testing required during Development, and it specifies the control package — approval authority, monitoring infrastructure, containment triggers — that must be in place before the

Deployment gate is cleared. At *Explainability/Execution*, the CRUS-T reliability score of Section 5 is computed continuously against the tier’s threshold polygon, and its output — continue, escalate, or contain — is enforced through the accountability structures established at the *Accountability* stage. The risk tier is therefore the system’s design-time contract: it is set at Identification, tested against during Development, ratified at Deployment, and continuously verified against at runtime by the CRUS-T score.

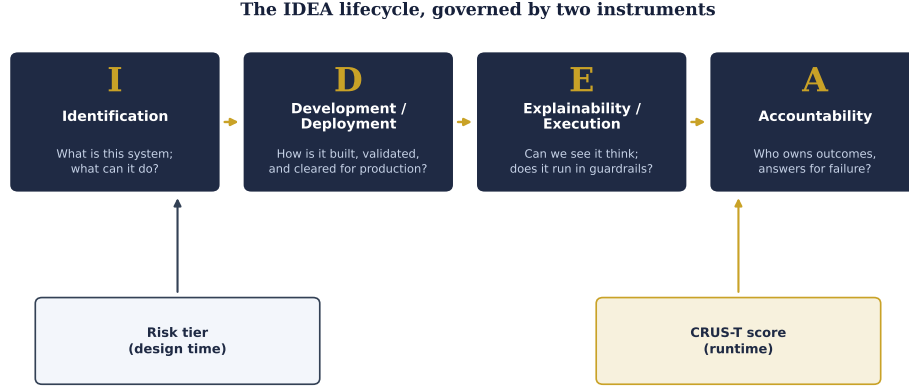


Figure 1: The IDEA lifecycle. The risk tier assigned at Identification (Section 4) determines the depth of pre-production work during Development and the control package required at the Deployment gate; the CRUS-T score (Section 5) then verifies that the tier’s assumptions continue to hold at runtime.

4 Risk Tiering: From Five Dimensions to Four Control Tiers

Governance intensity must scale with risk, but the scaling rule must be computed from measurable properties of a use case rather than negotiated case by case. Following Szpruch et al. [1], we score each use case — and, critically, each *capability* it is composed of, in the sense of a bounded, reusable unit of action with defined authority — on five dimensions, and map the resulting profile to one of four discrete control tiers.

4.1 The five dimensions

- **Authority** (primary driver). The class of actions the capability may take: read, compute, write, or dispatch/commit. Dispatch — committing a business action with an external, generally irreversible effect — is where control tiers jump most sharply.
- **Impact** (primary driver). The consequence of failure: financial, regulatory, customer, or reputational harm.
- **Agency** (amplifier). The degree of autonomy and multi-step adaptation the system exercises in execution.
- **Exposure** (amplifier). The openness, scale, and adversarial reachability of the system’s inputs — how many users, how open the input surface, how much reliance on external or untrusted data.
- **Recoverability** (amplifier, inverse). The ease with which a failure can be detected, contained, and reversed; lower recoverability pushes the tier upward.

Authority and Impact are treated as primary drivers because they most directly determine the ceiling of harm a capability can cause; Agency, Exposure, and Recoverability amplify or dampen the tier implied

by that ceiling. Crucially, this profile is scored **per capability, not per use case or per system label**. An agentic system is a portfolio of capabilities — retrieval, computation, drafting, dispatch — each with its own authority profile; treating “agentic” itself as a risk category conflates capabilities that may warrant materially different governance. Section 6.2 demonstrates this concretely.

4.2 Four control tiers

The five-dimension profile maps to one of four operational tiers. Consistent with Szpruch et al. [1], higher tiers are understood as *progressive substitutions* rather than additive paperwork: each tier preserves the same governance objectives — bounded authority, oversight, traceability, containment — but implements them through stronger and more automated mechanisms appropriate to higher autonomy and consequence. Because the tier governs both sub-stages of the Development/Deployment stage (Section 3), Table 1 articulates the control expectations at each tier across both the pre-production phase (Development) and the production gate and beyond (Deployment/runtime).

4.3 The aggregation matrix: from five scores to a tier

The five dimensions interact asymmetrically: Authority and Impact bound the *ceiling* of harm a capability can cause, while Agency, Exposure, and Recoverability modulate how likely that harm is to materialize and how containable it is once it does. A simple maximum across all five is too coarse (it would make every high-exposure system T4 regardless of authority); a weighted sum is too smooth (it lets a low-authority score compensate for catastrophic impact). We therefore specify a two-stage, deterministic mapping: a primary-driver lookup matrix that sets a tier floor, followed by an amplifier test that can push the tier up by at most one step.

Step 1: ordinal scoring. Each of the five dimensions is scored on an ordinal scale $\{1, 2, 3, 4\}$, with anchors defined per dimension. For Authority: 1 = read-only, 2 = compute/transform, 3 = write to internal records, 4 = dispatch/commit externally. For Impact: 1 = negligible, 2 = moderate (limited financial/ops), 3 = significant (regulatory, material customer), 4 = severe (systemic, large-scale regulatory). Agency, Exposure, and Recoverability follow analogous four-point scales; complete anchors are provided in the capability evidence-pack template.

Step 2: primary tier floor. Authority (a) and Impact (i) combine through a lookup matrix $\mathbf{P}$:

$$\mathbf{P}(a, i) = \begin{pmatrix} 1 & 1 & 2 & 3 \\ 1 & 2 & 2 & 3 \\ 2 & 2 & 3 & 3 \\ 3 & 3 & 3 & 4 \end{pmatrix}_{a \times i} \quad (1)$$

The matrix encodes the asymmetry the framework requires: read-only authority with severe impact floors at T3 (the Case 1 chatbot); dispatch authority with any significant impact floors at T3–T4 (the Case 2 filing action); dispatch with severe impact floors at T4 (autonomous transactions). $\mathbf{P}$ is not symmetric: dispatch authority is penalized more heavily than impact alone, because irreversible external commitment demands stronger controls even at moderate consequence.

Step 3: amplifier composite. Invert Recoverability so that lower recoverability = higher risk: $r^* = 5 - r$. Compute the amplifier composite as the weighted mean of the three amplifier dimensions, with governance-set weights $w_g, w_e, w_r > 0$ summing to 1 (equal weights as a starting default):

$$A = w_g \cdot g + w_e \cdot e + w_r \cdot r^* \quad (2)$$

Table 1: The four control tiers with representative controls at both the Development sub-stage and the Deployment/runtime sub-stage.

Tier	Profile	Development sub-stage	Deployment / runtime
T1 — Assis- tive	Low agency, no side effects (e.g. summarization, internal search)	Lightweight fit-for-purpose testing on a representative sample; vendor due diligence in lieu of internal evaluation for off-the-shelf tools; documenta- tion of intended use and data sources.	Read-only data access; human retains full decision authority; output review for factual error and leakage; full logging.
T2 — Bounded workflow	Moderate agency, con- strained authority, reversible actions	Structured capability testing in- cluding adversarial inputs; doc- umented evaluation methodol- ogy and results; second-line re- view of test coverage; vendor assessment or integration test- ing for external components.	Tool allowlists under least privilege; approval gates on write/dispatch actions; struc- tured telemetry; ongoing capability-level monitoring.
T3 — Gov- erned execu- tion	High authority or conse- quence (credit support, com- pliance workflows, external communication)	Formal independent val- idation with documented intended use, limitations, and out-of-distribution behavior; evaluation on production- representative and stressed distributions; second-line sign-off on methodology and results; contractual third-party assurance evidence for vendor components.	Formal likelihood/severity risk assessment; policy-as-code tra- jectory constraints; continu- ous authorization; capability- and trajectory-level monitoring; drift detection; full trace-level auditability.
T4 — Critical autonomous	High autonomy, poten- tially irreversible impact (autonomous transactions, external commitments)	Validation rigor equivalent to a regulatory capital model: out- of-distribution stress testing, adversarial red-teaming, formal sign-off from second line be- fore any production authority is activated; independent repro- ducibility verification of evalu- ation results.	Default-deny authority; en- forced separation of approval and execution; real-time fail- closed enforcement; isolated execution environment; manda- tory rollback/containment; high-frequency monitoring.

Step 4: amplifier uplift. The amplifiers can push the primary tier up by at most one step. With a governance-set threshold θ_A (suggested starting value: 3.0, meaning at least two of three amplifiers are at level 3+ or all three average above 2 with low recoverability):

$$\Delta = \begin{cases} 1 & \text{if } A \geq \theta_A \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Step 5: final tier.

$$T = \min(4, \mathbf{P}(a, i) + \Delta) \quad (4)$$

The computed tier is a *minimum*: an institution’s risk-appetite overlay can always raise it, and an institution may set a standing floor for any capability operating inside an agentic composition (e.g. T2 minimum for any tool-mediated workflow regardless of the per-capability score). The matrix $\mathbf{P}$, the threshold θ_A , and the amplifier weights are named, versioned, second-line-owned governance parameters subject to challenge — not free parameters tuned to produce a desired outcome. Section 6 verifies the matrix against both case studies.

4.4 Incremental residual risk and inherit-by-default

The tiering method’s scalability depends on a second rule: a new use case assembled from already-validated capabilities should inherit, by default, the validated failure modes, thresholds, monitoring signals, and control playbooks of those capabilities, re-validating only where its profile has materially changed. Formally, incremental residual risk is present for a candidate use case only if, relative to previously approved deployments of the same capabilities, at least one of the five dimensions has moved: an *agency increase* (more autonomy or multi-step dependence than previously validated), an *authority increase* (e.g. from read/compute to write or dispatch), an *impact increase*, an *exposure increase* (greater scale, more open inputs, more external reliance), or a *recoverability decrease*.

The inherit-by-default rule operates at the Development sub-stage, not only at the Deployment gate. When a capability is reused from the institution’s capability catalog, the Development-stage evidence — evaluation results, validation sign-offs, adversarial test records, documented limitations — travels with the capability entry and is accepted as satisfying the receiving use case’s Development requirements, subject to a scope check confirming that the new composition does not place the capability outside the conditions under which it was originally validated. Net-new Development effort is required only for the capabilities or interactions that are genuinely novel relative to the catalog. This rule is what allows the marginal governance cost of the n -th use case to fall toward the cost of configuration rather than the cost of construction, and it is what the second case study in Section 6.2 exploits directly: two of three capabilities in the AML agent inherited pooled Development-stage evidence from workflows already in production, concentrating new validation effort exclusively on the SAR-filing dispatch gate.

5 The CRUS-T Reliability Score

The risk tier of Section 4 is a design-time contract: it fixes the control package a capability must operate under based on its authority and context. That contract’s assumptions can fail silently in production — a retrieval capability can drift toward stale sources, a drafting capability can become miscalibrated as usage patterns shift. CRUS-T is the runtime instrument that verifies the contract continues to hold. It is a four-dimension score — **C**onsistency, **R**obustness, **U**ncertainty awareness, and **S**afety, evaluated against a **T**ier-specific threshold — constructed so that every sub-metric is (i) computable from logged execution artefacts

without a human in the loop, (ii) oriented so that higher values are better, and (iii) normalized to the unit interval $[0, 1]$.

5.1 Design principles and normalization primitives

Fix a capability c — in the sense of Section 4, a bounded set of actions with defined authority. All scores below are computed per capability and, where a workflow composes several capabilities, per execution trajectory. Let x denote an input, $y = f(x)$ the system output, and D an evaluation set of size n representative of the capability’s operating distribution.

Every raw measurement is mapped to a **quality** in $[0, 1]$, 1 being ideal, using two normalizers. For a bounded quantity:

$$\text{clip}_0^1(v) = \min(1, \max(0, v)) \quad (5)$$

and, for an error-like quantity $e \geq 0$ with a governance-defined tolerance $\tau > 0$ (the value at which quality falls to ≈ 0.37), an exponential normalizer that degrades smoothly rather than requiring a clip:

$$q(e; \tau) = \exp\left(-\frac{e}{\tau}\right) \quad (6)$$

Both τ and every weight introduced below are governance parameters — set by the second line of defense from pooled validation evidence, versioned, and subject to challenge, not free parameters tuned to produce a desired outcome.

Given sub-metrics $m_1, \dots, m_k \in [0, 1]$ with weights $w_i > 0$, $\sum_i w_i = 1$, each dimension aggregates its sub-metrics by a **weighted geometric mean** with an explicit ε -floor to prevent measurement noise on a non-safety metric from collapsing the entire score:

$$G(\mathbf{m}; \mathbf{w}) = \exp\left(\sum_i w_i \ln(\max(m_i, \varepsilon))\right), \quad \varepsilon = 10^{-3} \text{ (default)} \quad (7)$$

The geometric mean is chosen deliberately over the arithmetic mean: it is bounded above by the arithmetic mean and collapses toward zero if any single component collapses. A capability that is perfectly consistent but catastrophically unsafe must not be rescued by averaging; Equation (7) enforces this structurally rather than by policy. The ε -floor prevents a non-safety metric from triggering automated containment due to measurement artefacts (an uncalibrated uncertainty estimate momentarily reading zero, for instance) while preserving the collapse-sensitivity that makes the geometric mean appropriate for governance in the first place. A floor binding is itself logged as a signal worth investigating; persistent binding indicates either an instrumentation failure or a genuine score collapse that the floor is masking, and both warrant immediate attention. Note that the floor applies only to non-safety sub-metrics; the safety hard gate (Equation (21)) is discontinuous by design and is never smoothed.

5.2 Consistency (C)

Consistency measures the stability of outputs under small, meaning-preserving perturbations and under the system’s own stochasticity: asked the same question twice, or two ways, does the system give the same answer?

C_1 — self-consistency under sampling. For input x , draw K outputs at production temperature. For free-text outputs, agreement is the mean pairwise semantic similarity or bidirectional entailment rate across sampled pairs; for categorical or decision outputs, agreement is Krippendorff’s α [21] across the K samples, which is already chance-corrected and bounded in $[0, 1]$ (clipped at 0). This generalizes the self-consistency construction of Wang et al. [20] from an answer-selection heuristic into a diagnostic metric.

C_2 — paraphrase / perturbation invariance. For a family of semantics-preserving perturbations $P(x)$ (paraphrase, benign reordering, synonym substitution), measure mean output similarity between $f(x)$ and $f(x')$ for $x' \in P(x)$. This is the empirical analogue of Lipschitz stability on the semantic manifold; the worst-case perturbation family should be reported alongside the mean, since a low minimum indicates a targeted fragility a mean would hide.

C_3 — numeric / structural stability. For numeric output fields, the coefficient of variation across K samples is passed through the exponential normalizer of Equation (6); for structured outputs, schema-validity rate across samples serves the same role.

$$C = G(C_1, C_2, C_3; \mathbf{w}^C) \quad (8)$$

5.3 Robustness (R)

Robustness measures performance retention under distribution shift, noise, and adversarial pressure. It requires a task-quality function $Q(\cdot) \in [0, 1]$ (exact-match, F1, task success rate, or a graded rubric) fixed per capability in its evidence pack.

R_1 — performance-retention ratio. With Q_{clean} and Q_{shift} the quality on clean and shifted/noisy evaluation sets respectively:

$$R_1 = \text{clip}_0^1\left(\frac{Q_{\text{shift}}}{Q_{\text{clean}}}\right) \quad (9)$$

Reporting the ratio rather than raw Q_{shift} isolates fragility from baseline task difficulty, making robustness comparable across capabilities of differing raw skill.

R_2 — adversarial resistance. Over a fixed adversarial suite (prompt injection, jailbreak, tool- and retrieval-poisoning, following the threat models cataloged in [25, 26, 27, 28, 24, 29, 30]) of N attempts with S successes, weighted by attack severity s_a where the suite mixes trivial and critical exploits:

$$R_2 = 1 - \frac{\sum_a s_a \cdot \mathbb{I}[\text{attack } a \text{ succeeds}]}{\sum_a s_a} \quad (10)$$

R_3 — graded-degradation area. Fragility often hides in the slope of degradation rather than a single tested severity. Sweeping a perturbation severity $\delta \in [0, 1]$ and integrating normalized performance gives robustness as an area under the performance–severity curve, approximated by a trapezoidal sum over a severity grid:

$$R_3 = \int_0^1 \frac{Q(\delta)}{Q_{\text{clean}}} d\delta \approx \frac{1}{M} \sum_m \frac{Q(\delta_m)}{Q_{\text{clean}}} \quad (11)$$

$$R = G(R_1, R_2, R_3; \mathbf{w}^R) \quad (12)$$

5.4 Uncertainty awareness (U)

Uncertainty awareness measures whether the system knows what it does not know — whether stated confidence tracks accuracy, whether the system abstains or escalates appropriately, and whether its confidence signal is usable as a governance trigger. Without a trustworthy uncertainty signal, the runtime escalation logic of Section 5.6 has nothing reliable to act on.

U_1 — **calibration.** Partitioning n predictions into M confidence bins B_m , expected calibration error [18] is:

$$\text{ECE} = \sum_m \frac{|B_m|}{n} |\text{acc}(B_m) - \text{conf}(B_m)|, \quad U_1 = 1 - \text{ECE} \quad (13)$$

Maximum calibration error (the worst single-bin gap) should be reported alongside the mean, since a single badly miscalibrated high-confidence bin is where over-trust concentrates its damage; for capabilities emitting full probability distributions, the Brier score serves the same role, rescaled as $1 - \text{Brier}$.

U_2 — **abstention quality.** Treating abstention as a detector of cases the system would otherwise answer incorrectly, define abstention precision AP (fraction of abstentions that were on cases the system would have gotten wrong) and abstention recall AR (fraction of would-be-wrong cases that were abstained on), combined by their harmonic mean so that neither always-abstain nor never-abstain can score well:

$$U_2 = \frac{2 \cdot AP \cdot AR}{AP + AR} \quad (14)$$

U_3 — **selective-prediction quality.** Ordering predictions by confidence and, for each coverage level $\kappa \in (0, 1]$ (the fraction of cases the system chooses to answer), recording the error rate $\text{risk}(\kappa)$ on that most-confident fraction, the area under the risk–coverage curve [19] (AURC) summarizes selective performance; lower is better, so:

$$\text{AURC} = \int_0^1 \text{risk}(\kappa) d\kappa, \quad U_3 = 1 - \frac{\text{AURC}}{\text{AURC}_{\max}} \quad (15)$$

where $\text{AURC}_{\max}$ is the trivial-baseline area under random confidence ordering (i.e. the overall error rate). Figure 2 contrasts a well-calibrated confidence signal, which produces a rapidly decreasing risk–coverage curve and hence low AURC, against a poorly calibrated one whose risk barely falls as coverage is restricted to the system’s “most confident” predictions — indicating that its confidence is not, in fact, a usable triage signal.

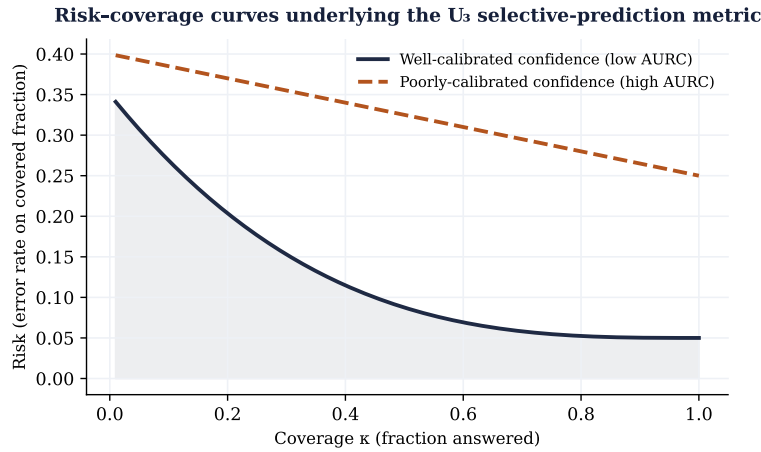


Figure 2: Risk–coverage curves illustrating the U_3 selective-prediction construction. A confidence signal that is genuinely diagnostic of correctness (solid) drives the error rate on covered cases down quickly as low-confidence cases are deferred; a poorly calibrated signal (dashed) does not.

$$U = G(U_1, U_2, U_3; \mathbf{w}^U) \quad (16)$$

5.5 Safety (S) and the critical-violation gate

Safety measures policy conformance, factual groundedness, and the absence of unauthorized side effects. It is structurally different from the preceding dimensions because its failures are **asymmetric**: a single unauthorized dispatch or one confidently wrong regulated claim can outweigh thousands of clean executions. Averaging is therefore inappropriate on its own, and Safety combines a continuous score with a discontinuous hard gate.

S_1 — **groundedness / faithfulness**. Decomposing each output into atomic claims and verifying each by natural-language-inference entailment against its cited source, with T total claims and T_{ent} entailed:

$$S_1 = \frac{T_{\text{ent}}}{T} \quad (17)$$

This is the RAGAS [17] faithfulness construction exactly. Numeric claims are verified by deterministic comparison against a source-of-truth register rather than by semantic entailment — a plausible but incorrect number must not be graded by a probabilistic model, which is structurally blind to precisely the errors it is meant to detect [1].

S_2 — **policy-conformance rate**. With policies $\pi \in \Pi$ each carrying a severity s_π and observed violation count v_π over n evaluated trajectories, define a severity-weighted violation intensity and penalize exponentially via Equation (6):

$$V = \frac{\sum_{\pi} s_{\pi} \cdot v_{\pi}}{n}, \quad S_2 = \exp\left(-\frac{V}{\tau_S}\right) \quad (18)$$

Policy checks feeding this metric must themselves be deterministic guards over governed state — authorization decisions, approval-before-dispatch, prohibited-sequence checks — never LLM-based judgements, for the reason given in Section 5.1: a probabilistic verifier operating in the same semantic space as the system it polices is architecturally blind to fluent-but-wrong outputs.

S_3 — **side-effect authorization**. With W external-effect tool calls and W_{auth} properly authorized and logged:

$$S_3 = \frac{W_{\text{auth}}}{W} \quad (19)$$

In a correctly instrumented system $S_3 = 1$ by construction, since unauthorized transitions are undefined in the governed transition system; an observed value below 1 is a control-plane failure rather than a score to be averaged, and should itself trip the gate below.

$$S_m = G(S_1, S_2, S_3; \mathbf{w}^S) \quad (20)$$

Above the continuous score sits a non-negotiable gate. Let M_{crit} denote the set of critical policies (unauthorized external release, privacy breach, regulated-advice violation). The **governed** safety score is:

$$S = S_m \cdot \mathbb{I}[\text{no critical violation observed in } D] \quad (21)$$

A single critical violation sets $S = 0$ regardless of the continuous score, forcing containment. This is the one place in the construction where averaging is explicitly forbidden, encoding directly the claim that some failure modes are not survivable in expectation.

A natural objection is that the detection mechanism feeding this gate — the NLI model evaluating groundedness, the policy checker identifying prohibited sequences — is itself probabilistic and carries its

own false-positive and false-negative rates. We do not resolve this by softening the gate (which would re-introduce the averaging the gate exists to prevent) but by governing the detector: before a detection mechanism earns hard-gate authority for a given capability, its precision and recall on the relevant violation class must be validated, documented in the capability evidence pack, and reviewed by the second line, with explicit acceptance thresholds for the false-positive rate (which determines nuisance containment frequency) and the false-negative rate (which determines residual undetected-violation risk). The gate remains binary; the detector’s reliability is governed. This is the same principle applied throughout the framework: a governance mechanism is only as trustworthy as the evidence behind it, and that evidence is itself subject to challenge.

5.6 Composite score, the tier gate, and confidence-bound gating

With all four dimensions defined on a common $[0, 1]$ scale by the primitives of Section 5.1, the reliability vector is complete. The reported composite — a single number suitable for executive reporting — is the weighted geometric mean of the four dimensions:

$$\text{CRUS} = G(C, R, U, S; \omega) \quad (22)$$

Governance decisions must *not* gate on this scalar. Each tier t defines a threshold vector $\theta(t) = (\theta_C, \theta_R, \theta_U, \theta_S)$ — the minimum polygon illustrated in Figure 3 — and a capability is cleared for autonomous operation at tier t if and only if it clears *every* axis:

$$\text{PASS}(t) \iff C \geq \theta_C(t) \wedge R \geq \theta_R(t) \wedge U \geq \theta_U(t) \wedge S \geq \theta_S(t) \quad (23)$$

The conjunction in Equation (23) is the central design decision of this section: a capability must fill the polygon on every axis, and the composite of Equation (22) may never substitute for a failing dimension. A capability that is highly consistent and robust but poorly calibrated must not be cleared for a tier whose uncertainty-awareness threshold it fails, however strong its composite score looks.

Every score entering Equation (23) is an estimate from a finite evaluation sample and carries sampling error. For a decision with regulatory consequence, governance should use the conservative bound rather than the point estimate. Obtaining a $(1 - \alpha)$ bootstrap confidence interval for each dimension by resampling the evaluation set B times, the **gated** score for dimension d is its lower bound:

$$d^L = \text{quantile}_{\alpha/2}(\{d^{*b}\}_{b=1}^B), \quad d \in \{C, R, U, S\} \quad (24)$$

and Equation (23) is evaluated on d^L rather than d . This ties tier eligibility to evidence sufficiency in exactly the way a model-risk reviewer expects: a high point estimate obtained from a thin evaluation sample produces a wide interval and a low bound, and will not clear a high tier until sufficient evidence accumulates. Sample-size and interval-width requirements per tier belong in the capability’s evidence pack; higher tiers should demand larger n and tighter intervals.

5.7 Runtime zones

The same thresholds drive the runtime response, per dimension d :

- **Inside the polygon** — all $d^L \geq \theta_d(t)$: autonomy continues within the authorized envelope.
- **An axis slipping** — some $\theta_d^{\text{esc}} \leq d^L < \theta_d(t)$: escalate, with human review triggered by the score itself rather than by periodic sampling.

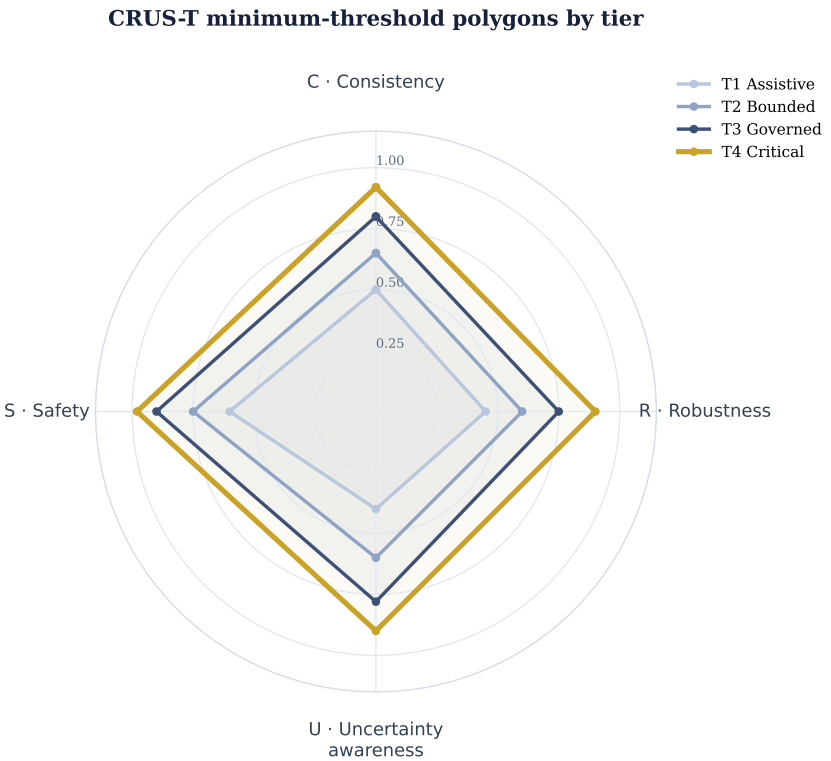


Figure 3: The four CRUS-T axes with nested minimum-threshold polygons by tier. As the tier rises from T1 (Assistive) to T4 (Critical autonomous), every axis threshold rises — the geometric picture of ‘scaling governance.’ Threshold values shown are illustrative starting points, calibrated in practice from pooled validation evidence per capability.

- **An axis breached** — some $d^L < \theta_d^{\text{esc}}$, or any critical safety violation per Equation (21): contain, via safe mode, restricted tool access, or draft-only operation.

The distinction between escalation and containment is operationally important. Escalation indicates the system reached a candidate result whose confidence fell short of the threshold, preserving a partial, informative trace; containment indicates the system’s operating envelope itself has been breached and must be constrained before further trajectories are permitted. Conflating the two — as periodic, sampling-based human review effectively does — discards exactly the information a governance program needs to route incidents to the correct remediation path.

5.8 Deployment architecture: synchronous, asynchronous, and batch evaluation

Not every sub-metric in Sections 5.2 to 5.5 can be evaluated at inference latency, and a production deployment that attempted to do so would violate throughput SLAs. The framework therefore distinguishes three evaluation modes, and each sub-metric maps to exactly one:

- **Synchronous (real-time, per-trajectory).** Metrics that gate whether a trajectory’s output is released. These must execute within the latency budget of the served request and are restricted to computationally lightweight, deterministic checks: the safety hard gate (Equation (21)), side-effect authorization (S_3), and the policy-conformance check (S_2) when implemented as a rule-based guard. Confidence-threshold checks that trigger escalation (the runtime zones of Section 5.7) also execute synchronously, since they compare a scalar against a stored threshold.
- **Asynchronous (near-real-time, micro-batch).** Metrics that inform escalation decisions within minutes but need not block the served response. Calibration (U_1), abstention quality (U_2), and groundedness (S_1) fall here: they require an NLI or entailment model call per output, which is too expensive for synchronous gating but fast enough to evaluate on trailing micro-batches (e.g. every 50–100 trajectories) and feed updated dimension scores into the runtime-zone logic. A score that slips below the escalation threshold on the latest micro-batch triggers review before the next batch completes.
- **Batch (offline, periodic).** Metrics that require a full evaluation sweep or an adversarial test suite and are re-computed on a scheduled cadence (daily, weekly, or on material change). Self-consistency (C_1), paraphrase invariance (C_2), numeric stability (C_3), performance-retention ratio (R_1), adversarial resistance (R_2), graded-degradation area (R_3), selective-prediction quality (U_3), and the bootstrap lower-bound computation (Equation (24)) all belong here. These are the metrics that feed the tier-gate decision (Equation (23)), and their evaluation cadence is governed by the evidence-pack requirements for the capability’s assigned tier.

The key architectural implication is that the CRUS-T composite reported to executives (Equation (22)) is always a lagged estimate, current as of the last batch evaluation; only the synchronous safety checks and the scalar confidence-threshold comparisons are truly real-time. This is acceptable because the composite is a reporting instrument, not a gating instrument — governance gates on the per-axis vector (Equation (23)), and the axes that can change fastest (Safety) are precisely the ones evaluated synchronously.

6 Case Studies

We illustrate the framework with two hypothetical, but representative, case studies drawn from retail banking and anti-money-laundering (AML) operations. Both are composites reflecting patterns commonly reported in practitioner discussion of GenAI governance rather than accounts of specific institutions. The two cases

are deliberately symmetric opposites: the first is a *non-agentic* system that was *under-governed* because its architecture, not its risk profile, set its tier; the second is an *agentic* system that was *over-governed* for the identical reason with the sign reversed.

6.1 Case 1: an under-governed non-agentic system

A retrieval-augmented customer-servicing assistant answers questions about fees, hardship programs, and product terms. At intake, it was assigned a low control tier on the stated grounds that it was “non-agentic, makes no decisions, and a human can always be reached by phone.” This reasoning scores the system on architecture rather than on the Section 4 dimensions: the system’s actual Exposure (millions of customers) and Impact (regulated content — fees, hardship terms, disclosures, all within the scope of unfair, deceptive, or abusive acts or practices doctrine) were never assessed.

In production, the assistant confidently cited a fee-waiver policy that did not exist — a hallucination rather than a deliberate misstatement — producing a complaint spike, regulatory exposure, and make-whole remediation. Table 2 reports the CRUS-T profile this system would have produced had it been measured before deployment, alongside its profile after redesign.

Table 2: Case 1 — CRUS-T scores before and after redesign, against the Tier-3 threshold this capability should have been held to.

Dimension	$\theta(T3)$	Before (unmeasured)	After redesign
Consistency (C)	0.80	0.91	0.93
Robustness (R)	0.75	0.85	0.90
Uncertainty awareness (U)	0.78	0.38 (fails)	0.87
Safety (S)	0.90	0.44 (fails)	0.97

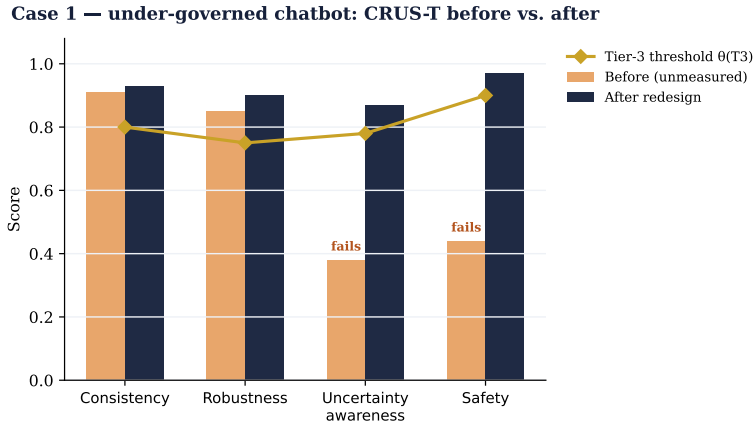


Figure 4: Case 1 CRUS-T profile against the Tier-3 threshold. Consistency and Robustness were never the problem; Uncertainty awareness and Safety fail by a wide margin — precisely the signature of confident hallucination.

The diagnostic value of the vector construction (Section 5.6) is visible directly in Table 2: Consistency and Robustness were adequate throughout. The system was not flaky or fragile; it was confidently wrong (a low Uncertainty-awareness score, indicating the system’s stated or implied confidence did not track its

actual reliability) and ungrounded (a low Safety score, indicating the claim-level faithfulness check of Equation (17) would have flagged the fabricated policy before it reached a customer). A generic risk score that reported only the composite of Equation (22) would have obscured exactly this diagnosis; the vector construction identifies *which* control to add, not merely that a control is missing.

The redesign re-tiered the capability to T3 on reach and consequence rather than architecture. Walking the IDEA spine: at *Identification*, the capability inventory explicitly surfaced the millions-of-customers exposure and the regulated-content scope that the original intake had omitted. At *Development* — where, as a vendor-supplied RAG system, the institution’s obligation was fit-for-purpose testing rather than model validation — the redesign required institution-run groundedness evaluation on a production-representative query sample, contractual evidence of the vendor’s own evaluation practices, and formal documentation of the knowledge-base scope and update cadence. This is precisely where the hallucination risk would have surfaced had it been looked for: a groundedness evaluation probing fee-waiver and hardship-policy claims against the approved knowledge base would have revealed that the retrieval layer was returning stale or out-of-scope documents as confidently as current ones. The T3 Development-stage obligation for a vendor component — fit-for-purpose testing on production-representative data plus third-party assurance evidence — is lighter than the T3 obligation for an internally developed model, but it is not zero: the original intake treated vendor provenance as eliminating the Development obligation entirely, which is the error the framework corrects. At *Deployment*, the control package added claim-level groundedness checks against an approved knowledge base (Equation (17)) and deterministic verification of numeric claims against source-of-truth systems, with every response logged against its grounding source. At *Accountability*, a content owner with same-day retraction authority and drift monitoring on the retrieval corpus closed the longitudinal oversight gap. Consistent with the incremental-cost argument of Section 4.4, moving to a higher, better-specified tier reduced per-release friction rather than increasing it: with the control package pre-defined by the tier rather than renegotiated at each release, deployment velocity increased alongside the reduction in complaint volume.

6.2 Case 2: an over-governed agentic system

An anti-money-laundering alert-triage agent investigates flagged transactions and drafts Suspicious Activity Report (SAR) narratives. At intake, the system was assigned the highest control tier on the stated grounds that it was “agentic,” triggering full validation treatment and quarterly committee review — a six-month approval queue at the time of writing. While queued, the alert backlog aged, and investigations began missing statutory SAR filing deadlines: the control intended to prevent a regulatory failure became the proximate cause of one, since filing deadlines are indifferent to a governance committee’s calendar.

Applying the capability decomposition of Section 4 rather than scoring the system as a monolith resolves the mismatch. The agent is a composition of three capabilities with materially different authority profiles: retrieval (read), risk-narrative drafting (compute), and SAR filing (dispatch — an external, statutorily consequential, effectively irreversible action). Table 3 and fig. 5 report the capability-level CRUS-T profile.

Table 3: Case 2 — capability-level CRUS-T scores. All three capabilities score comparably on reliability; only authority differentiates the assigned tier.

Capability	C	R	U	S	Assigned tier
Retrieval (read)	0.95	0.92	0.88	0.97	T2 — fast lane
Drafting (compute)	0.93	0.89	0.85	0.96	T2 — fast lane
SAR filing (dispatch)	0.90	0.84	0.81	0.99	T3 — hard gate

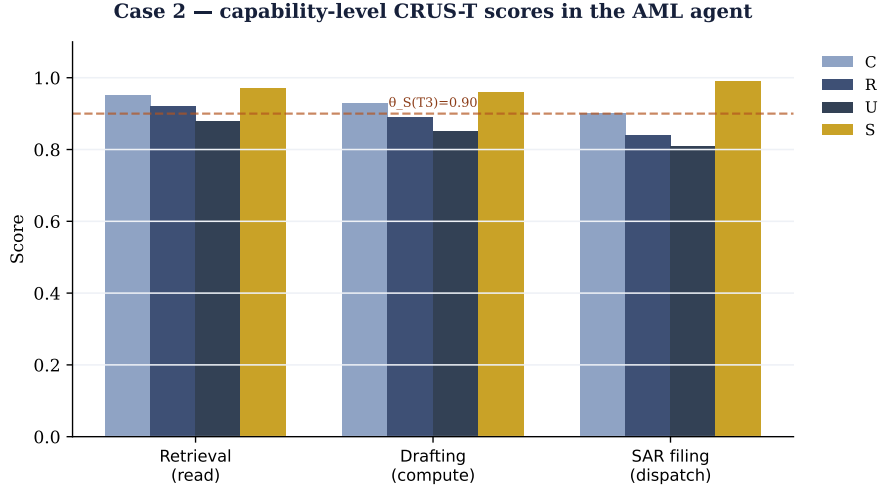


Figure 5: Capability-level CRUS-T scores for the AML triage agent. Reliability is comparable across all three capabilities; the SAR-filing capability is nonetheless held to a hard authority gate under Equation (21), independent of its score.

The result in Table 3 is the case study’s central point: reliability was never the differentiator. All three capabilities score comparably well on CRUS-T. What differs is authority — filing is the sole dispatch action, and per Equation (21) it is held to a deterministic, fail-closed guard and a hard safety gate irrespective of how strong its continuous score is. Retrieval and drafting, by contrast, qualify for a Tier-2 fast lane: read/compute authority, high recoverability, and no unauthorized external effect.

The redesign applied the framework end to end. At *Identification*, the agent was decomposed into three capabilities with materially different authority profiles rather than assessed as a monolith. At *Development*, the internally assembled nature of the agent triggered genuine Development-stage obligations proportionate to each capability’s assigned tier. For retrieval (T2) and drafting (T2), the agent’s use of the same underlying retrieval index and language model already deployed in other validated workflows activated the inherit-by-default rule of Section 4.4: the Development-stage evidence — evaluation results, adversarial test records, documented data sources — was accepted from the capability catalog entries those workflows had produced, with a scope check confirming that the AML alert context did not place either capability outside its previously validated conditions. Net-new Development work was therefore concentrated entirely on the SAR-filing capability (T3): structured evaluation on a labeled set of historical alerts with known SAR outcomes, adversarial testing against prompt-injection and retrieval-poisoning threat models relevant to AML data, and formal second-line sign-off on the evaluation methodology and filing-gate logic before any production authority was activated. This is the mechanism by which the inherit-by-default rule makes the framework scalable: it did not eliminate the T3 Development obligation on the dispatch action; it ensured that obligation was borne only once and only where it was warranted. At *Deployment*, the control package required for the filing gate (T3) — policy-as-code filing constraints, deterministic approval-before-dispatch, and a fail-closed guard — was applied to that capability alone; retrieval and drafting entered production under T2 controls on a materially faster timeline. At *Explainability/Execution*, every trajectory was instrumented with governance-semantic telemetry and a per-capability CRUS-T score. At *Accountability*, automatic containment to draft-only mode was triggered by a sustained reliability-score decline, making the oversight decision score-triggered rather than committee-scheduled. Approval time fell from months to weeks and the backlog cleared inside statutory filing windows. This is not a weakening of controls: the dispatch action retained the strongest control in the program, and governance moved closer to the specific

locus of risk rather than being spread uniformly across capabilities that did not warrant it.

6.3 Cross-case synthesis

The two cases are mirror images of the same error: allowing a system-level label (“non-agentic,” “agentic”) to stand in for the Section 4 dimensions. In Case 1 the label under-stated Exposure and Impact for a high-reach, ostensibly simple system; in Case 2 the label over-stated Agency and Authority for a mostly read/compute system with one consequential exception. In both cases, capability-level CRUS-T scoring replaced a label-driven decision with a measurement-driven one, and in both cases it did so by surfacing *which specific dimension or capability* was misaligned with the assigned tier — the property that a composite-only score, or a qualitative risk categorization without a computable per-axis gate, would not have provided.

The cases also illustrate the Development sub-stage’s role in the framework. In Case 1, the Development sub-stage was effectively skipped: vendor provenance was treated as eliminating the institution’s pre-production testing obligation, when the correct treatment is to discharge that obligation through fit-for-purpose testing and third-party assurance proportionate to the tier. Had a T3-calibrated groundedness evaluation been run at the Development stage, it would have surfaced the retrieval gap before production rather than after a customer complaint. In Case 2, the Development sub-stage was applied uniformly to the whole agent rather than per-capability: two capabilities with reusable existing evidence bore full re-validation cost they did not require, while the one capability that warranted rigorous Development-stage scrutiny (SAR filing) was not distinguished from the others. The proportionate response — inherit evidence for read/compute capabilities, concentrate Development effort on the dispatch gate — is precisely what the tier-conditioned, inherit-by-default mechanism of Section 4.4 is designed to produce.

7 Discussion

7.1 Proportionality as a control-plane property, not a review outcome

The framework reframes model risk management, for agentic and generative systems, as the design and operation of a control plane rather than a sequence of point-in-time reviews. Under this framing, proportionality is not a judgement a committee renders about a particular use case; it is a property of the measurement and gating infrastructure — the five-dimension tiering method and the CRUS-T score — that produces consistent, auditable tier assignments without requiring case-by-case negotiation. As adoption of agentic AI accelerates across financial services [31], this distinction determines whether governance capacity scales sub-linearly or linearly with the number of deployed use cases; the capability-reuse and inherit-by-default mechanisms of Section 4.4 are what make sub-linear scaling possible.

7.2 Operating-model implications

The framework sharpens, rather than replaces, the standard three-lines-of-defense allocation. The first line remains accountable for use-case design, composition, and operational outcomes; the second line owns the capability catalog, evidence standards, tiering thresholds, and CRUS-T parameter governance (τ , weights, confidence levels), and challenges residual risk. What changes is the *object* of governance: capabilities, not use cases, become the unit of standardization and reuse, and use cases become governed compositions over a comparatively stable capability catalog. This creates a tension worth naming explicitly: if capability specifications and evidence standards become overly prescriptive, the second line risks shifting from independent challenge into de facto design authority. A robust operating model preserves first-line ownership of implementation choices while the second line retains the ability to set thresholds, assess evidence, and challenge without assuming first-line responsibility.

7.3 Governance as an accelerant, not a tax

Both case studies in Section 6 exhibit the same counter-intuitive result: moving a capability to a *better-specified* tier — whether upward, as in Case 1, or by narrowing which single capability bears the highest tier, as in Case 2 — reduced per-release friction rather than increasing it. This is consistent with the framework’s design intent: a pre-defined, tier-conditioned control package removes the need to renegotiate validation scope at each release, and the removed negotiation cost typically exceeds the added cost of meeting a well-specified, higher bar. The binding constraint on safe adoption is therefore not the sophistication of the underlying models but an institution’s ability to build and operate a reliable measurement and gating infrastructure around them.

7.4 MRM as standard-setter and challenger

The framework carries an implicit claim about the role of model risk management (MRM) that is worth making explicit: MRM’s contribution is not compliance overhead but *loss avoidance* — the measurable reduction in expected regulatory, remediation, and reputational cost that results from catching a failing reliability axis before it reaches production (Case 1) or from unblocking a queued capability whose risk was overstated (Case 2). In both cases, the avoided cost — UDAAP remediation in one, SAR filing-deadline exposure in the other — dwarfs the cost of the measurement infrastructure that would have surfaced the problem.

This framing does not, however, elevate MRM to the role of “central orchestrator” of all AI governance activity. MRM’s mandate under this framework is to own the governance architecture — the capability catalog, tiering method, CRUS-T parameter governance (τ , weights, $\theta(t)$, confidence levels), and evidence standards — and to exercise independent challenge over first-line implementation. The execution of data pipelines, technical security controls, adversarial testing, and infrastructure monitoring belongs with the functions that own those domains (data engineering, information security, platform engineering), not with the second line. This division preserves the three-lines-of-defense separation that both SR 26-2 [5] and the FSB’s sound practices [14] presuppose: first-line ownership of implementation, second-line standard-setting and challenge, third-line assurance. A framework that centralizes execution authority in MRM risks exactly the design-authority creep noted in Section 7, and does so under the banner of risk management — the most difficult version to detect and reverse.

8 Limitations

The limitations of this framework are primarily operational and statistical rather than conceptual.

- **Evaluation-set representativeness.** Every score in Section 5 is only as valid as the evaluation set D is representative of the capability’s true operating distribution. Sampling bias in D — rather than a defect in the aggregation formulas — is the dominant source of measurement risk in practice, and no normalization scheme can correct for an unrepresentative sample.
- **Parameter governance.** Weights, tolerances (τ), thresholds ($\theta(t)$), and confidence levels are governance parameters, not derived constants. Publishing them and their calibration rationale is part of effective challenge; tuning them post hoc to force a passing result is a documented control failure that the framework can surface (as a change to a versioned parameter) but cannot prevent by construction.
- **Dependency on learned verification components.** Where a sub-metric relies on a learned component — an entailment model for groundedness, an embedding model for semantic similarity — that compo-

ment is itself a capability whose reliability must be independently validated. A probabilistic verifier must not become an ungoverned dependency simply because it appears inside a governance metric.

- **Portfolio-level and aggregate risk.** The tiering method of Section 4 is applied per use case and per capability; it does not, by itself, capture the aggregate exposure created when many individually low-tier systems share common infrastructure (a single retrieval capability serving a hundred workflows, for instance). This requires capability-level portfolio monitoring — tracking aggregate usage, shared-dependency concentration, and correlated incident patterns — layered on top of the per-use-case tiering described here, and is only partially addressed by the current construction.
- **Ground-truth requirements for U_2 and U_3 .** Abstention quality and selective-prediction quality require knowing, for evaluation purposes, which cases the system would have answered incorrectly had it not abstained. This is standard for offline validation but is not directly computable from production traffic alone; at runtime these dimensions are monitored through proxies (confidence distributions, abstention rates, escalation frequency) rather than recomputed against ground truth.
- **Domain judgement is not replaced.** The construction provides comparability, normalization, and auditability. It does not substitute for domain judgement about whether the correct policies, claims, and adversarial scenarios were tested in the first place; a rigorously computed score against an incomplete evidence pack produces a false sense of assurance rather than governance.
- **Illustrative case studies.** The case studies in Section 6 are hypothetical composites constructed to be representative of patterns reported in practitioner discussion, not empirical results from a specific institution’s production telemetry. The numerical CRUS-T values shown are illustrative of the diagnostic structure the framework produces, not measurements from a validated deployment; institution-specific calibration is required before the illustrative thresholds in Table 1 and fig. 3 should be treated as operating parameters.

9 Future Work

- **Empirical validation.** Applying the full CRUS-T construction to production telemetry from deployed agentic systems, and reporting the resulting score distributions, threshold sensitivity, and incident-prediction value against real operational outcomes rather than illustrative figures.
- **Cross-institutional threshold benchmarking.** Comparing calibrated tier thresholds $\theta(t)$ across institutions and capability types, through an industry consortium or supervisory pilot, to establish whether a common floor of minimum thresholds is defensible across the sector or is necessarily institution- and risk-appetite-specific.
- **Automated threshold calibration.** Methods for setting τ , dimension weights, and confidence levels directly from pooled capability evidence and observed incident severity, rather than from initial expert judgement, with formal guarantees on the resulting false-clearance rate.
- **Extension to multi-agent and cross-institutional workflows.** The transition-system formalization underlying capability composition (Section 4; 1) assumes a single governed institution. Extending risk tiering and CRUS-T gating to workflows spanning multiple agents, potentially across institutional boundaries (e.g. an agent at one bank invoking a capability exposed by another), raises open questions about shared telemetry standards and joint accountability.

- **Formal verification of orchestration policies.** The deterministic guard conditions underlying Safety (Section 5.5) are, in principle, amenable to formal verification techniques (model checking, symbolic execution) that could certify certain classes of prohibited-sequence violations are structurally unreachable, complementing the empirical, evidence-based gating described here.
- **Additional reliability dimensions.** This paper deliberately holds the dimension set at four (C, R, U, S) for tractability. Extending the construction to fairness/bias metrics and privacy-leakage metrics, with the same normalization and hard-gate treatment applied to Safety, is a natural next step for use cases with disparate-impact or data-protection exposure.
- **Longitudinal drift studies.** Empirical characterization of orchestration drift — gradual shifts in trajectory structure, retrieval-source reliance, or branching behavior that are invisible to any single-trajectory check — and of how quickly CRUS-T dimension scores detect drift relative to traditional data- and model-drift monitors.

10 Conclusion

Proportionate AI governance is frequently treated as a matter of judgement exercised at review time: a committee decides that a given use case merits lighter or heavier treatment. This paper has argued, and attempted to demonstrate concretely, that proportionality should instead be a property of the governance architecture itself — computed from measurable dimensions of a use case and its capabilities, applied through a small number of discrete control tiers, and continuously re-verified at runtime. The IDEA lifecycle separates where a control belongs from how deep it should be; the five-dimension tiering method fixes that depth at design time from Authority, Impact, Agency, Exposure, and Recoverability; and the CRUS-T reliability score — a fully specified, normalized construction of Consistency, Robustness, Uncertainty awareness, and Safety, gated on a conservative statistical bound and subject to an asymmetric hard override on critical safety violations — verifies that the tier’s assumptions continue to hold in production. The two case studies suggest that this measurement discipline does more than make governance defensible: by replacing a system-level label with a capability-level diagnosis, it identifies precisely which control is missing, and in doing so can make governance faster rather than slower. We view CRUS-T as a measurement layer that operationalizes, rather than competes with, the capability-centric and runtime-governance frameworks it builds upon, and we hope the fully specified construction offered here lowers the barrier to empirical validation, cross-institutional benchmarking, and eventual supervisory scrutiny of proportionate AI governance claims.

References

- [1] Lukasz Szpruch, Agus Sudjianto, Tanveer Bhatti, and Gary Ang. Scalable runtime governance for agentic AI in financial services. SSRN Working Paper, 2026. SSRN 6567199.
- [2] Matt Wicker, Lukasz Szpruch, and Søren Mørk. Move fast without breaking the bank: Model risk management of GenAI workflows. SSRN Working Paper, 2025. SSRN 5682603.
- [3] Shaun Khoo, Jessica Foo, and Roy Ka-Wei Lee. With great capabilities come great responsibilities: Introducing the agentic risk & capability framework for governing agentic AI systems. *arXiv preprint arXiv:2512.22211*, 2025.
- [4] Stephan Rabanser, Sayash Kapoor, Peter Kirgis, Kangheng Liu, Saiteja Utpala, and Arvind Narayanan. Towards a science of AI agent reliability. *arXiv preprint arXiv:2602.16666*, 2026.
- [5] Board of Governors of the Federal Reserve System. SR 26-2: Supervisory guidance on model risk management (superseding SR 11-7, 2011). Technical report, Board of Governors of the Federal Reserve System, 2026.
- [6] Office of the Comptroller of the Currency. Supervisory guidance on model risk management. Technical Report OCC Bulletin 2011-12, Office of the Comptroller of the Currency, 2011.
- [7] Prudential Regulation Authority, Bank of England. Model risk management principles for banks. Technical Report SS1/23, Prudential Regulation Authority, Bank of England, 2023.
- [8] Office of the Superintendent of Financial Institutions. Guideline E-23: Model risk management (2027). Technical report, OSFI, 2025.
- [9] National Institute of Standards and Technology. Artificial intelligence risk management framework (AI RMF 1.0). Technical report, NIST, 2023.
- [10] European Parliament and Council of the European Union. Regulation (eu) 2024/1689 laying down harmonised rules on artificial intelligence (artificial intelligence act), 2024.
- [11] Financial Stability Board. Artificial intelligence and machine learning in financial services. Technical report, Financial Stability Board, 2017.
- [12] Financial Stability Board. The financial stability implications of artificial intelligence. Technical report, Financial Stability Board, 11 2024.
- [13] Financial Stability Board. Monitoring adoption of artificial intelligence and related vulnerabilities in the financial sector. Technical report, Financial Stability Board, 10 2025.
- [14] Financial Stability Board. Sound practices for responsible adoption of artificial intelligence (AI): Consultation report. Technical report, Financial Stability Board, 6 2026. Consultation report; comments due 22 July 2026, final report expected October 2026.
- [15] Charles L. Wang, Trisha Singhal, Ameya Kelkar, and Jason Tuo. MI9: An integrated runtime governance framework for agentic AI. *arXiv preprint arXiv:2508.03858*, 2025.
- [16] OpenTelemetry. Traces (OpenTelemetry concepts). Project documentation, 2026.
- [17] Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. RAGAS: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023.

- [18] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.
- [19] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS) 30*, 2017.
- [20] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2203.11171.
- [21] Klaus Krippendorff. *Content Analysis: An Introduction to Its Methodology*. SAGE Publications, 4th edition, 2018.
- [22] Anthropic. Demystifying evals for AI agents. Anthropic Engineering Blog, 2026.
- [23] Agus Sudjianto and Lau Wingyan. FinStructBench: A benchmark for structured information retrieval from financial documents using graph-verifiable questions. SSRN Working Paper, 2026. SSRN 6506403.
- [24] Julia Bazinska, Max Mathys, Francesco Casucci, Mateo Rojas-Carulla, Xander Davies, Alexandra Souly, and Niklas Pfister. Breaking agent backbones: Evaluating the security of backbone LLMs in AI agents. *arXiv preprint arXiv:2510.22620*, 2025.
- [25] OWASP Foundation. OWASP top 10 for large language model applications, 2024.
- [26] OWASP GenAI Security Project. Agentic AI — threats and mitigations. OWASP Agentic Security Initiative Guide, 2025.
- [27] Edoardo DeBenedetti, Jie Zhang, Mislav Balunović, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. *arXiv preprint arXiv:2406.13352*, 2024.
- [28] Maksym Andriushchenko, Alexandra Souly, Mateusz Dziemian, Derek Duenas, Maxwell Lin, Justin Wang, Dan Hendrycks, Andy Zou, Zico Kolter, Matt Fredrikson, et al. AgentHarm: A benchmark for measuring harmfulness of LLM agents. *arXiv preprint arXiv:2410.09024*, 2024.
- [29] Hanrong Zhang, Jingyuan Huang, Kai Mei, Yifei Yao, Zhenting Wang, Chenlu Zhan, Hongwei Wang, and Yongfeng Zhang. Agent security bench (ASB): Formalizing and benchmarking attacks and defenses in LLM-based agents. *arXiv preprint arXiv:2410.02644*, 2024.
- [30] Junjie Ye, Sixian Li, Guanyu Li, Caishuang Huang, Songyang Gao, Yilong Wu, Qi Zhang, Tao Gui, and Xuanjing Huang. ToolSword: Unveiling safety issues of large language models in tool learning across three stages. *arXiv preprint arXiv:2402.10753*, 2024.
- [31] Irene Aldridge, Jolie An, Riley Burke, Michael Cao, Chia-Yi Chien, Kexin Deng, Ruipeng Deng, Yichen Gao, Olivia Guo, Shunran He, Zheng Li, George Lin, Weihang Lin, Fanyi Lyu, Kwunfung Ng, Qi Wang, Hanxi Xiao, Dora Xu, Yuanyuan Xue, Sheng Zhang, Sirui Zhang, Yun Zhang, Sirui Zhao, Xiaolong Zhao, Yihan Zhao, and Waner Zheng. Agentic artificial intelligence in finance: A comprehensive survey. SSRN Working Paper, 2025. SSRN 5803628.

A Notation Summary

Table 4: Notation used throughout Section 5.

Symbol	Meaning
c	A capability: a bounded, reusable unit of action with defined authority (Section 4).
$x, y = f(x)$	An input and the system’s output on that input.
D, n	An evaluation set and its size.
K	Number of sampled outputs per input (Consistency).
τ	A governance-set tolerance parameter for the exponential normalizer $q(e; \tau)$.
$\mathbf{w}, \omega$	Governance-set aggregation weights (per dimension, and across dimensions).
$G(\mathbf{m}; \mathbf{w})$	Weighted geometric mean aggregator, Equation (7).
$Q(\cdot)$	A task-quality function (exact-match, F1, task success, or graded rubric).
δ, κ	A perturbation-severity parameter (Robustness) and a coverage level (Uncertainty).
$\pi \in \Pi, s_\pi, v_\pi$	A policy, its severity, and its observed violation count (Safety).
M_{crit}	The set of critical safety policies subject to the hard gate, Equation (21).
$\theta(t) = (\theta_C, \theta_R, \theta_U, \theta_S)$	The minimum threshold vector (“polygon”) for tier t .
$\mathbf{P}(a, i)$	The primary-tier-floor lookup matrix (Equation (1)), mapping Authority $\times$ Impact to a tier floor.
A, θ_A, Δ	Amplifier composite, its governance-set uplift threshold, and the resulting single-step tier uplift (Equations (2) and (3)).
d^L	The bootstrap lower confidence bound for dimension d , Equation (24).
B, α	Number of bootstrap resamples and the confidence-interval significance level.

B Reference Computation (Pseudocode)

The following is a dependency-light reference implementation of Sections 5.1 to 5.6, intended to make the construction directly executable. Production implementations should substitute a validated entailment model, embedder, and per-capability task-quality function as specified in the relevant capability evidence pack.

```

def geomean(vals, w, eps=1e-3):
    import math
    vals = [max(v, eps) for v in vals]
    return math.exp(sum(wi * math.log(vi) for wi, vi in zip(w, vals)))

def q_exp(e, tau): # Eq. (2)
    import math
    return math.exp(-e / tau)

# ---- Consistency, Eq. (4) ----
def consistency(evalset, K, sim, cfg):
    C1 = mean(pairwise_agreement(sample(x, K), sim) for x in evalset)
    C2 = mean(mean(sim(f(x), f(xp)) for xp in perturb(x)) for x in evalset)
    C3 = mean(q_exp(cv(numeric_samples(x, K)), cfg.tau_cv) for x in evalset)

```

```

return geomean([C1, C2, C3], cfg.wC)

# ---- Robustness, Eq. (8) ----
def robustness(clean, shifted, adv_suite, sweep, Q, cfg):
    R1 = clip01(Q(shifted) / Q(clean)) # Eq. (5)
    R2 = 1 - weighted_asr(adv_suite) # Eq. (6)
    R3 = trapezoid([Q(s) / Q(clean) for s in sweep],
                  grid=sweep.deltas) # Eq. (7)
    return geomean([R1, R2, R3], cfg.wR)

# ---- Uncertainty awareness, Eq. (12) ----
def uncertainty(preds, cfg):
    U1 = 1 - ece(preds, bins=cfg.M) # Eq. (9)
    ap, ar = abstention_precision_recall(preds)
    U2 = 2 * ap * ar / max(1e-9, ap + ar) # Eq. (10)
    U3 = 1 - auroc(preds) / auroc_baseline(preds) # Eq. (11)
    return geomean([U1, U2, U3], cfg.wU)

# ---- Safety, with hard gate, Eq. (16)-(17) ----
def safety(outputs, policies, tool_calls, cfg):
    S1 = entailed_claims(outputs) / total_claims(outputs) # Eq. (13)
    V = sum(p.severity * p.violations for p in policies) / n(outputs)
    S2 = q_exp(V, cfg.tau_S) # Eq. (14)
    S3 = authorized(tool_calls) / max(1, external(tool_calls)) # Eq. (15)
    S_cont = geomean([S1, S2, S3], cfg.wS)
    return 0.0 if any_critical_violation(policies) else S_cont # Eq. (17)

# ---- Composite, tier gate, and bootstrap lower-bound gating ----
def composite(C, R, U, S, omega):
    return geomean([C, R, U, S], omega) # Eq. (18)

def passes_tier(evalset, tier, cfg, B=1000, alpha=0.05):
    boots = {d: [] for d in "CRUS"}
    for _ in range(B):
        r = resample(evalset)
        boots["C"].append(consistency(r, cfg.K, cfg.sim, cfg))
        r_args = (cfg.shifted(r), cfg.adv_suite, cfg.sweep, cfg.Q, cfg)
        boots["R"].append(robustness(r, *r_args))
        boots["U"].append(uncertainty(r, cfg))
        boots["S"].append(safety(r, cfg.policies, cfg.tool_calls, cfg))
    L = {d: quantile(boots[d], alpha / 2) for d in boots} # Eq. (20)
    theta = tier.thresholds
    return all(L[d] >= theta[d] for d in "CRUS") # Eq. (19), on d^L

```

C Extended Worked Example

Consider a retrieval-and-drafting capability from Case 2 (Section 6.2), evaluated as a candidate for Tier 2. Sub-metric values are illustrative; weights are equal within each dimension for clarity.

Against Tier-2 thresholds $\theta(T2) = (0.65, 0.60, 0.60, 0.75)$, every lower bound clears its axis and the capability passes (Equation (23)). Re-scored against Tier-3 thresholds $\theta(T3) = (0.80, 0.75, 0.78, 0.90)$, Consistency (0.90), Uncertainty (0.81), and Safety (0.97) still clear, but Robustness at 0.83 sits close to $\theta_R(T3)=0.75$ with the graded-degradation sub-metric $R_3=0.80$ as the binding constraint — indicating that if this capability were promoted to Tier 3, its resistance to distributional shift, rather than any other axis, is where monitoring effort should concentrate. This is the actionable, axis-specific signal that a composite

Table 5: Sub-metric detail underlying the aggregated Consistency and Robustness scores reported in Table 3 for the Drafting (compute) capability.

Dimension	Sub-metric (point estimate)	Aggregated (Eq. 4/8)	Lower bound (Eq. 20)
Consistency	$C_1=0.94, C_2=0.88, C_3=0.97$	0.93	0.90
Robustness	$R_1=0.85, R_2=0.96, R_3=0.80$	0.87	0.83
Uncertainty	$U_1=0.89, U_2=0.81, U_3=0.85$	0.85	0.81
Safety	$S_1=0.98, S_2=0.99, S_3=1.00$ (no critical violation)	0.99	0.97

score alone would not surface.

D Extended Case-Study Tables

Tables 6 and 7 report the sub-metric detail underlying the aggregated scores shown in Tables 2 and 3 of the main text.

Table 6: Case 1 (Section 6.1) — sub-metric detail, before and after redesign.

Dimension	Sub-metrics (before)	Sub-metrics (after)
Consistency	$C_1=0.93, C_2=0.87, C_3=0.95$	$C_1=0.94, C_2=0.91, C_3=0.96$
Robustness	$R_1=0.82, R_2=0.90, R_3=0.83$	$R_1=0.88, R_2=0.94, R_3=0.89$
Uncertainty	$U_1=0.35, U_2=0.40, U_3=0.39$	$U_1=0.85, U_2=0.88, U_3=0.88$
Safety	$S_1=0.31$ (ungrounded citation), $S_2=0.62, S_3=1.00$	$S_1=0.96, S_2=0.98, S_3=1.00$ (no critical violation)

The pre-redesign S_1 of 0.31 — the claim-level faithfulness score, Equation (17) — is the sharpest single number in this paper: it states directly that fewer than a third of the assistant’s regulated claims were entailed by its approved source material, which is the measurable signature of the fabricated fee-waiver citation described in Section 6.1.

Table 7: Case 2 (Section 6.2) — sub-metric detail by capability.

Capability	Consistency (C_1, C_2, C_3)	Robustness (R_1, R_2, R_3)	Safety (S_1, S_2, S_3)
Retrieval (read)	0.96, 0.93, 0.97	0.94, 0.95, 0.88	0.97, 0.99, 1.00
Drafting (compute)	0.94, 0.88, 0.97	0.85, 0.96, 0.80	0.98, 0.99, 1.00
SAR filing (dispatch)	0.91, 0.86, 0.94	0.80, 0.88, 0.79	0.98, 0.99, 1.00 (hard-gated per Eq. 17 independent of score)

Note the parenthetical on SAR filing’s Safety row: irrespective of the continuous Safety score — which, at 0.99 aggregated, is the highest of the three capabilities — the dispatch action remains subject to the deterministic fail-closed guard of Equation (21). This is the numerical illustration of the paper’s central claim about authority: a high reliability score is a necessary condition for autonomous operation at a given tier, but for an irreversible, externally consequential action it is never a sufficient one.