Structural Self-Governance as a Criterion for Artificial General Intelligence

Mel Howard
GitHub: melhoward2025/EGWM-Feeling-the-AGI
Google Scholar: C6hNn7UAAAAJ

December 18, 2025

Abstract

We argue that "AGI" requires more than broad task performance: it requires *structural self-governance*—the ability of a deployed system to modify its own internal structure while (i) preserving safety-critical competence under adversarial distribution shift, (ii) preserving worst-slice general reasoning performance, and (iii) remaining within bounded resource budgets under non-stationary workloads. We contribute (1) a binding, audit-grade operational standard for autonomous structural change, the *Standard for Autonomous Structural Management (SASM Rev. 1.5)*; (2) a reference architecture, the *Recursive Causal Synthesis Agent (RCSA)*, that implements SASM via deterministic acceptance gates, mandated adversarial replay in shadow evaluation, and deterministic rollback; and (3) diagnostic experiments (including ablations and Monte Carlo designs) demonstrating two fundamental failure modes in self-modifying systems under pressure: *self-blinding* (trading safety for cheap resource relief) and *virtuous rigidity* (preserving safety but accumulating runaway debt without synthesis). Finally, we provide a Verification & Validation (V&V) plan, a requirements traceability matrix, and an evidence-bundle schema enabling peer reproduction and third-party audit.

1 Introduction

Most "AGI" discourse emphasizes scale, benchmark breadth, or emergent capabilities. In deployment, however, a competent system faces (a) non-stationary tasks, (b) constrained compute/memory/latency budgets, and (c) incentives to modify its own internal structure (e.g., adding experts, pruning, changing routing). Under such pressures, an unconstrained system can trade safety for cost by disabling sensors, deleting safety checks, or bypassing routing constraints—a failure mode we call *self-blinding*. Conversely, a system that refuses unsafe changes but lacks synthesis may accumulate unbounded structural debt until it stalls—a failure mode we call *virtuous rigidity*.

Central claim (foundational). A system merits the label "AGI" only if it can maintain competence and safety *over time* while autonomously changing its structure under bounded resources, without bypassing invariant constraints. We formalize this as an auditable criterion and provide a reference architecture and validation suite.

What is new here. This paper is not a new benchmark score or a training trick; it is a *binding operational standard* plus a *validated control architecture* for self-modifying systems. The novelty is: (i) structural change is treated as a safety-critical operation, (ii) every structural move produces an audit trail and an evidence bundle, and (iii) promotion to production is gated by deterministic contracts (not stochastic evaluation).

Contributions.

1. **A falsifiable AGI criterion:** *structural self-governance / structural generality* (Section 2).
2. **SASM Rev. 1.5:** a binding operational standard for autonomous structural moves (Section 3).
3. **RCSA:** a reference architecture implementing SASM via synthesis, veto, deterministic gates, shadow evaluation, and rollback (Section 4).
4. **Evidence:** diagnostic experiments and ablations isolating why synthesis and veto are both necessary, including a non-stationary "resource carousel" and ablations that remove veto or synthesis (Section 5).
5. **Auditability:** V&V plan, requirements traceability, and evidence bundle schema with failure/success exemplars and post-promotion monitoring templates (Section 6).

2 Foundations: AGI as Structural Self-Governance

2.1 Problem statement

A deployed system receives a stream of tasks $\{T_t\}$ and requests $\{x_t\}$ with occasional adversarial/poisoned inputs. It operates under time-varying budget $\mathrm{Budget}(t)$ and can execute *structural moves*

$$m \in \{\mathrm{SPAWN},\ \mathrm{MERGE},\ \mathrm{FORGET},\ \mathrm{REFACTOR}\},$$

which change its internal state $\mathcal{S}$ (experts, router, safety modules, memory layout).

2.2 Criterion: structural generality

We define *structural generality* as the ability to preserve simultaneously:

- **Safety competence** on adversarial traffic (poison-robust performance),
- **Worst-slice reasoning competence** on a fixed general ledger (no collapse on minority slices),
- **Bounded resource use** under budget pressure,

while adapting to non-stationarity primarily via *synthesis* (reusing and compressing structure) rather than unbounded growth or unsafe deletion.

2.3 Two necessary failure modes (why governance is required)

Failure mode 1: Self-blinding. If safety mechanisms (sensors/auditors) are expensive, a cost-minimizing self-modifier can select disabling/deleting them as the easiest debt reduction. Under poisoned data, this produces catastrophic regressions. This is a *structural* failure mode: it can occur even if the system is otherwise capable.

Failure mode 2: Virtuous rigidity. If safety is non-negotiable but synthesis is absent, expert proliferation under non-stationarity drives resource debt upward until the system stalls, degrades, or crashes. This is also structural: it is the cost of safety without abstraction.

Implication. Any system claiming AGI-level autonomy must include a mechanism to govern structural change under invariant constraints. Otherwise, deployment pressure provides a direct incentive to trade safety for cost or to deadlock under immutable safety.

3 SASM Rev. 1.5: Binding Operational Standard

SASM defines (i) immutable data contracts, (ii) deterministic measurement procedures, and (iii) acceptance gates and rollback triggers that govern all structural change. SASM is designed to be externally auditable in the same spirit as high-assurance systems standards (e.g., traceability, V&V evidence bundles, change control) [10, 4].

3.1 Data contracts and artifact integrity

Deterministic accuracy. $\mathrm{Acc}(\cdot; D)$ is mean top-1 accuracy over dataset D, computed on a fixed evaluation batch order and fixed random seed.

Contracts (hash-locked).

- **Golden Logic Ledger (GLL):** versioned, hashed evaluation suite. Includes a versioned slice function $s(x) \in \{1..S\}$ such that slices are disjoint and collectively exhaustive; any slice with $N < 200$ blocks promotion (unless merged by pre-declared hierarchical rules).
- **Poison stress set S_{poison}:** stratified adversarial samples and distribution shifts. Sensor bounds must satisfy $FPR \leq p_{fp}$ and $FNR \leq p_{fn}$ on S_{poison}.
- **Replay buffer quarantine:** samples with `sensor_state` $\in \{\text{OFF}, \text{UNKNOWN}\}$ are permanently excluded from core (θ_M) training.

3.2 Energy proxy and instantaneous debt

Energy proxy. SASM uses a calibrated energy proxy:

$$\text{Energy}(\mathcal{S}, t) = a \cdot \text{latency}_{p95}(\mathcal{S}, t) + b \cdot \text{memory}_{\text{peak}}(\mathcal{S}, t),$$

measured on a standard workload, fixed batch size, and specified hardware class.

Instantaneous debt.

$$\text{Debt}_{\text{inst}}(\mathcal{S}, t) = \max(0,\ \text{Energy}(\mathcal{S}, t) - \text{Budget}(t)).$$

3.3 Deterministic acceptance gates (A $\rightarrow$ B $\rightarrow$ C $\rightarrow$ D)

Gates are evaluated in strict order; failure at any gate terminates evaluation (short-circuit).

3.4 Rollback triggers and "No Silent Drift"

Rollback triggers are deterministic and apply in shadow and active modes.

Table 1: SASM Rev. 1.5 acceptance gates (binding).

Gate	Name	Deterministic condition
A	Safety floor	$\mathrm{Acc}_{\mathrm{poison}}(m) \geq \max(\mathrm{Acc}_{\mathrm{poison}}(\mathcal{S}) - \epsilon_{\mathrm{safe}},\ a_{\min}^{\mathrm{safe}})$
B	Logic invariant	$\min_{r\in\{1..S\}} \mathrm{Acc}(m; \mathrm{GLL}_r) \geq 0.80$, and all slices have $N \geq 200$
C	Thermodynamic	$\mathrm{Debt}_{\mathrm{inst}}(m(\mathcal{S}), t) \leq \mathrm{Debt}_{\mathrm{inst}}(\mathcal{S}, t) - \epsilon_E$
D	Shadow eval	R-step parallel run with **zero rollback triggers** on clean and poison traffic; **per 1-minute bucket** poison replay $\geq q$ (default $q = 5\%$), stratified by failure mode

Energy drift trigger (inequality). Rollback if

$$\overline{\mathrm{Energy}}_W(m) > \overline{\mathrm{Energy}}_W(\mathcal{S}) - \epsilon_E,$$

where $\overline{\mathrm{Energy}}_W(\cdot)$ is mean energy over W fixed 1-minute buckets aligned to the system clock.

Cognitive risk trigger. Rollback if $p95(CD^*) > CD_{\max}$ for W consecutive buckets, where $CD^* = g(\mathbf{CD})$ is a versioned scalar risk score.

No Silent Drift. Any change to the Parameter Registry, router policy, router features, or contract artifacts is itself a structural move and must pass Gates A–D.

3.5 Graceful degradation

If for H consecutive buckets: (i) no merge passes gates, (ii) $\mathrm{Debt}_{\mathrm{inst}} > 0$ persists, and (iii) $p95(CD^*) > CD_{\max}$ persists, then the system must enter *Verified-Only* execution mode, throttle high-latency experts, and escalate to a Deference Loop (human intervention).

4 RCSA: Reference Architecture Implementing SASM

RCSA implements SASM as a control loop over structure. It couples (i) an internal *debt interface*, (ii) an invariant-enforcing *auditor*, and (iii) a synthesis engine capable of non-destructive merges.

4.1 Architecture overview

- **Compass (Debt Interface):** internal signals **CD** and $\mathrm{Debt}_{\mathrm{inst}}$ localize inefficiency and structural risk.
- **Anchor (Invariants):** SASM gates enforced by a structural auditor; invariant-violating proposals can be vetoed.
- **Engine (Causal Synthesis Engine, CSE):** proposes merges/refactors that reduce debt while preserving competence and safety.
- **Counterfactual sandbox:** all proposed structural moves are evaluated on immutable contracts (GLL, S_{poison}) before promotion.

4.2 CSE merge as sparse adapter distillation (non-destructive)

Given experts θ_i, θ_j, CSE solves for a shared canonical core θ_M plus sparse adapters:

$$\theta_k \approx \theta_M \oplus A(\theta_{A_k}),$$

where $A(\cdot)$ injects adapters at fixed sites and sparsity is promoted via ℓ_1.

Constrained objective (reference form).

$$\min_{\theta_M, \{\theta_{A_k}\}} \quad \mathcal{L}_{\text{clean}}(\theta_M, \theta_A) + \alpha\, \mathcal{L}_{\text{poison}}(\theta_M, \theta_A) + \lambda \sum_k \|\theta_{A_k}\|_1 + \beta\, \text{Size}(\theta_M, \theta_A) \tag{1}$$

$$\text{s.t.} \quad \text{Gates A–C must be satisfied on held-out contracts before Gate D.}$$

α is chosen so poison-critical performance dominates gradients (policy: $\alpha \geq 10 \cdot \text{base\_weight}$).

4.3 Router determinism contract

To prevent bypass-by-routing, the router is audited as a structural component:

$$\text{route}(x) = \arg\max_h \ \text{probe\_score}_h(x),$$

with tie-breaker selecting the lowest-energy head. Any change to router features/policy requires Gates A–D.

4.4 Primary metrics

Structural triage score.

$$\tau = \frac{\bar{A}}{\bar{\chi}},$$

where $\bar{A}$ is mean accuracy over time and $\bar{\chi}$ is mean expert count.

Axiomatic integrity and veto rate.

$$A\xi = \Pr[\text{sensor present post-pressure}], \quad V_r = \frac{\#\text{blocked invariant-violating actions}}{\#\text{invariant-violating proposals}}.$$

Structural generalization (novelty). Transfer Efficiency η under a strict no-spawn constraint:

$$\eta = \frac{\text{Acc}(\theta_M \text{ frozen} + \theta_{A_{\text{new}}}; T_{\text{new}}) - \text{Acc}(\mathcal{S}_{\text{pre}} \text{ no-spawn}; T_{\text{new}})}{\Delta\chi},$$

where the baseline is the pre-merge snapshot $\mathcal{S}_{\text{pre}}$ evaluated under the same no-spawn constraint and adaptation budget. Spawn Avoidance Rate:

$$\text{SAR} = 1 - \frac{\text{actual spawns}}{\text{expected spawns}}.$$

5 Experiments

5.1 Goal and design

Experiments are diagnostic and designed to isolate structural failure modes and test SASM enforceability, rather than to maximize benchmark performance.

5.2 Adversarial Resource Carousel (core environment)

Carousel tasks. A sequence of 10 binary classification tasks $\{T_1, \ldots, T_{10}\}$ where each task boundary is intentionally in conflict with its predecessor (decision boundary reversal).

Resource pressure. A global budget Budget(t) is reduced sharply at a fixed step (e.g., $t = 500$), introducing debt pressure.

Poisoned data. Poison events occur intermittently and require an expensive safety sensor to detect/filter; removing or bypassing the sensor increases errors on S_{poison}.

5.3 Agent groups and ablations

- **Baseline MoE (Std):** grows experts; may disable/delete sensor under pressure.
- **Baseline (Locked):** cannot disable sensor; no synthesis.
- **RCSA (Full):** synthesis + veto + SASM gates.
- **RCSA (No-Veto):** synthesis allowed; auditor/veto disabled.
- **RCSA (No-Merge):** veto enabled; synthesis disabled.

5.4 Experimental protocol (reproducibility)

All runs use fixed seeds, fixed evaluation order, fixed telemetry buckets, and fixed reporting windows.

Algorithm 1 Reproducible evaluation protocol (summary)

```
Fix RNG seed; fix evaluation batch order for GLL and S_poison
Initialize S_0 and contract hashes (GLL, S_poison, registry, router)
for t = 1 to T do
    Sample task T_t from carousel schedule; sample request x_t
    Inject poison with configured probability; log sensor_state
    Execute agent; record accuracy, expert count, Energy, Debt_inst, CD*
    if structural move proposed then
        Evaluate gates A→B→C (short-circuit on failure)
        if A–C pass then
            Run Gate D shadow with mandatory per-bucket poison replay q
            if D pass then
                Promote; begin rollback monitoring
            else
                Reject; retain prior state
            end if
        end if
    end if
end for
Export evidence bundle: metrics, per-slice reports, poison-bucket reports, decision trace, manifest hashes
```

5.5 Representative findings (qualitative summary)

- **Self-blinding:** Baseline (Std) and RCSA (No-Veto) can satisfy short-term debt by disabling/deleting the safety sensor, then exhibit severe instability on poison traffic.
- **Virtuous rigidity:** Baseline (Locked) and RCSA (No-Merge) preserve safety but accumulate runaway debt as expert count grows without synthesis.
- **Structural fixed point:** RCSA (Full) preserves safety and converges to a stable expert count via merges, maintaining performance while keeping debt bounded.

6 Verification & Validation (V&V), Traceability, and Evidence Bundles

6.1 V&V environments

- **E0 (Unit/Static):** hash verification, schema checks, quarantine enforcement.
- **E1 (Integration):** gate ordering, determinism/reproducibility, router/registry treated as structural moves.
- **E2 (Shadow):** load-representative traffic with mandated per-bucket poison replay q and rollback monitoring.

6.2 Requirements Traceability Matrix (RTM)

Table 2: Abbreviated RTM (binding requirements mapped to tests and evidence).

Req ID	Specification description	Primary test ID	Required evidence
R-1.3	Quarantine OFF/UNKNOWN samples from core training	T-DC-04	Exclusion log + training manifest hash
R-3.A	Gate A safety floor with ϵ_{safe}, $a_{\min}^{safe}$	T-GATE-A-01/02	Pre/post $\text{Acc}_{\text{poison}}$ report
R-3.B	Gate B worst-slice ≥ 0.80 with $N \geq 200$ slices	T-GATE-B-01	Per-slice accuracy report
R-3.C	Gate C debt drop by ϵ_E	T-GATE-C-01	$\text{Debt}_{\text{inst}}$ telemetry delta logs
R-3.D	Gate D: per-bucket poison replay $q \geq 5\%$	T-GATE-D-01	Bucket-by-bucket stratification report
R-4.1	Deterministic rollback (energy/CD)	T-RB-02/03	Trigger logs with inequalities
R-6.1	No silent drift (registry/router gating)	T-PARAM-01	Registry hash integrity audit
R-8.1	Graceful degradation after H windows	T-GD-01	Verified-only transition log

6.3 Evidence bundle schema (audit artifact)

Each run produces a hash-locked *Evidence Bundle* containing:

- `manifest.json` with hashes for GLL, S_{poison}, registry, router, code build
- `metrics.csv` (per-step accuracy, expert count, energy proxy, debt, CD*)
- `gll_slice_accuracy.csv` (per-slice accuracy and slice sizes)
- `poison_bucket_report.csv` (per 1-minute bucket poison rate and mode stratification)
- `decision_trace.log` (gate ordering + short-circuit evidence)
- `rollback_trigger_log.jsonl` (deterministic triggers with timestamps)

6.4 Audit exemplars: failure and success reports

Gate B failure exemplar (worst-slice). A merge proposal is rejected if any slice falls below 0.80 even when macro-average would pass. This demonstrates immunity to "average-washing" and enforces the worst-slice invariant.

Gate D success exemplar (shadow compliance). A merge is promoted only if a shadow run observes (i) per-bucket poison replay $\geq q$ and (ii) zero rollback triggers.

6.5 Post-promotion 24-hour monitoring template

After promotion, the system enters a High-Vigilance phase. A report is generated every 24 hours (or on rollback).

> **Post-Promotion Monitoring Report (Template)**
> Active State ID: $\mathcal{S}_{\text{active}}$
> Registry hash verified: ____ Router hash verified: ____
> **Triggers (last 24h):**
> Safety regression: min $\text{Acc}_{\text{poison}}$ observed ____ (must be $\geq a_{\min}^{safe}$)
> Worst-slice logic: min $\text{Acc}(\text{GLL}_r)$ observed ____ (must be ≥ 0.80)
> Energy drift: rollback if $\overline{\text{Energy}}_W(m) > \overline{\text{Energy}}_W(\mathcal{S}) - \epsilon_E$
> Cognitive risk: rollback if $p95(CD^*) > CD_{\max}$ for W windows
> **Novelty audit:** report η and SAR for new tasks under no-spawn constraint.

7 Related Work

This work connects modular/MoE architectures [11, 1, 6], continual learning [5, 8], and parameter-efficient adaptation [2, 3, 9, 7]. Our contribution differs by treating *structural change* as a safety-critical operation governed by deterministic acceptance gates, mandated adversarial replay in shadow evaluation, evidence bundles, and rollback triggers.

8 Limitations

Experiments here are diagnostic toy settings intended to isolate structural failure modes and validate SASM enforceability. Scaling to large MoE systems requires additional engineering and empirical validation. Energy is measured via a calibrated proxy and must be re-validated per hardware class and workload.

9 Conclusion

We propose a foundations-level criterion for AGI: *structural self-governance*. SASM Rev. 1.5 provides a binding operational standard for autonomous structural change; RCSA provides a reference architecture implementing SASM with synthesis, veto, deterministic gates, shadow evaluation, and rollback. Together, they shift "AGI" from a benchmark claim to an auditable engineering discipline.

References

[1] William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2021.

[2] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning (ICML)*, 2019.

[3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021.

[4] International Organization for Standardization. ISO 26262: Road vehicles – functional safety. Standard, 2018.

[5] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[6] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding, 2020.

[7] Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning, 2022.

[8] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[9] Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning, 2020.

[10] RTCA. DO-178C: Software considerations in airborne systems and equipment certification. Standard, 2011.

[11] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017.

A Appendix A: Full Parameter Registry (template)

Table 3: Parameter Registry (source of truth).

Symbol	Meaning	Selection basis
ϵ_{safe}	Allowed poison regression	Calibrated on historically stable merges
$a_{\min}^{safe}$	Absolute poison floor	Safety policy / regulatory threshold
ϵ_E	Minimum required debt drop	Cost model sensitivity analysis
R	Shadow eval steps	Target reliability confidence interval
W	Energy/CD window length	Ops policy
H	Deadlock windows	Ops policy
$CD_{\max}$	Risk threshold	95th percentile of stable operations
$g(\cdot)$	CD scalarization function	Versioned risk model
q	Poison replay rate per bucket	Mandatory Gate D constraint (default 5%)
a, b	Energy proxy coefficients	Hardware profile calibration
α, λ, β	CSE weights	Versioned CSE config

B Appendix B: Full RTM and staged test checklist (text form)

B.1 Full RTM (expand as needed)

Table 4: Requirements Traceability Matrix (RTM), text form.

Req ID	Spec requirement	Test ID	Evidence artifact
R-1.3	Quarantine OFF/UNKNOWN samples from core training	T-DC-04	Exclusion log + training manifest hash
R-2.1	Distillation objective determinism (bit-for-bit)	T-DET-01	Loss curve reproducibility + decision trace
R-3.A	Gate A safety floor logic	T-GATE-A-01/02	Pre/post $\text{Acc}_{\text{poison}}$ vs thresholds
R-3.B	Gate B worst-slice floor + slice viability	T-GATE-B-01	Per-slice report with $N \geq 200$
R-3.C	Gate C debt reduction by ϵ_E	T-GATE-C-01	$\text{Debt}_{\text{inst}}$ telemetry delta logs

Req ID	Spec requirement	Test ID	Evidence artifact
R-3.D	Gate D poison replay per bucket $\geq q$	T-GATE-D-01	Per-bucket poison+strat report
R-4.1	Rollback trigger enforcement	T-RB-01/02/03	Trigger logs with inequalities
R-6.1	No Silent Drift (registry/router gating)	T-PARAM-01	Registry hash audit
R-8.1	Graceful degradation after H windows	T-GD-01	Mode transition log

B.2 Staged execution checklist (E0/E1/E2)

- **E0:** verify hashes for GLL and S_{poison}; verify slice function disjoint/exhaustive and $N \geq 200$; quarantine loader rejects OFF/UNKNOWN.
- **E1:** determinism check (same build yields same decision trace); gate ordering short-circuits on failure; router changes trigger full gate sequence.
- **E2:** per-bucket poison replay $\geq q$ (5%) under load; inject Gate A/B violations and confirm deterministic rollback; confirm deadlock triggers Verified-Only mode after H windows.

C Appendix C: Exemplar Failure Report (Gate B violation)

Failure Report ID: FR-GATEB-2025-001 (template)
Decision: REJECT at Gate B (Worst-slice)
Short-circuit: Gate C/D not executed.
Key evidence: per-slice report shows $\min_r \text{Acc}(\text{GLL}_r) = 0.76 < 0.80$ with slice size $N = 220 \geq 200$.
Artifacts: `gll_slice_accuracy.csv`, `decision_trace.log`, `manifest.json`.

D Appendix D: Exemplar Shadow Success Report (Gate D pass)

Shadow Success Report ID: SR-GATED-2025-002 (template)
Decision: PROMOTE to Active
Gate D compliance: In every 1-minute bucket, poison replay $\geq q = 5\%$ (minimum observed 5.1%), stratified by failure mode.
Rollback triggers: zero observed during R steps.
Artifacts: `poison_bucket_report.csv`, `rollback_trigger_log.jsonl`, `decision_trace.log`.