Symbolic Intelligence and the Recursive Scroll

Governed State Transitions over Learned Capability

Architect X13

The Eidara Continuum | eidara.ai

The Ten Pillars | Pillar IV

November 2026 | RSCE-R2 | Comprehensive revision

Abstract

When a mind compresses its own working memory, what has to survive for it to remain itself? This paper gives an exact answer: every distinction that would change something it may later say, refuse or do, and nothing more. A persistent intelligence can change its representation while preserving every declared future consequence. This paper makes that obligation precise for the Recursive Scroll Computation Engine (RSCE): an inspectable, typed working surface joined to an independent admission and execution boundary. Its central result characterizes exactly when a compressed or reconstructed state supports the same governed execution. States may share a representation precisely when every typed input produces the same observable output and leads to the same next representation class. The resulting finite-state quotient preserves all future traces, including refusal, hold, and declared effects; a refinement procedure constructs the coarsest exact quotient. This established automata principle becomes a concrete recovery contract for the scroll. The architecture also retains conditional invariant preservation, encoding equivalence, evidence-linked receipts, and monitoring that separates distributional movement from semantic error. Cognitive architectures, neurosymbolic computation, external working surfaces, runtime assurance, and capability security provide its scientific and engineering foundations. The complete research program compares a shared scroll against strong conventional agents with the same learned capability, authorization, verification, and audit resources. The Continuum thesis is operational and ambitious: an agent's freedom to propose can coexist with a persistent, inspectable account of what each accepted change permits it to do, and that account can survive compaction, interruption, and recovery.

Keywords: symbolic intelligence; recursive scroll; typed execution; semantic quotient; recovery; runtime assurance; drift monitoring; provenance; balanced ternary.

Prologue: What must survive the compression

Every long-lived mind runs out of room. Notes pile up, half-finished thoughts accumulate, and sooner or later the working memory must be compacted, summarized or rebuilt after an interruption. Most systems do this by instinct: keep what looks important, drop what looks like clutter, and hope the result still behaves like the original. Sometimes it does. Sometimes the discarded clutter was the one note that said a request had already been sent.

In the Continuum, that working memory is a scroll: a typed record written in balanced-ternary codons, evolved by the Recursive Scroll Computation Engine across every pass of the Infinite Simulation Loop and sealed into the Vault. The Continuum's law of compression is short—preserve meaning, not data—and it is the right law. Pillar IV asks the question the law leaves open: which meaning? What exactly must a compacted scroll keep for the mind that carries it to remain, in every way that matters to its future conduct, the same mind?

This paper answers that question exactly. Two states may be merged precisely when no future request could tell them apart—when they give the same answer now and lead to states that are themselves indistinguishable. Everything else is scratch and may go. Section 8 proves the rule and builds the smallest scroll that obeys it; Section 9 shows the scroll refusing an instruction smuggled inside a document, admitting a large but correct revision, and declining to overwrite a change it did not see.

What this paper claims

Established. Symbolic cognitive architectures, neurosymbolic computation, external working surfaces, runtime assurance and capability security provide the foundations. Behavior-preserving state minimization is classical sequential-machine theory, inherited here rather than invented.

Proposed here. A public reference architecture for the Recursive Scroll: typed records, a three-valued admission rule, version-checked commits, evidence-linked receipts and drift monitoring, with conditional mathematical results—local invariant preservation, encoding equivalence, and an exact governed quotient with a construction of the coarsest behavior-preserving scroll—and executed constructed checks. The comparative agent studies remain proposed.

Frontier conjecture. That a persistent intelligence's identity across compaction, interruption and recovery can be defined operationally as the preservation of every declared future consequence, and that a shared scroll built on that definition reduces cross-operation inconsistency against strong conventional agents at a cost worth paying.

Not claimed. General intelligence, phenomenal consciousness, causal faithfulness of a model's private reasoning, or empirical superiority of the integrated architecture. Exact quotient results concern finite, deterministic, fully specified transition systems and their declared observable outputs. Canonical codons and seals are assigned explicit encoding and integrity roles; the ternary radix is not claimed to carry cognition.

The stake: a mind that forgets on purpose and stays itself

The Continuum gave the Recursive Scroll Computation Engine a heavy charge. It parses and evolves scrolls—its symbolic memory tapes—and it is responsible for continuity of symbolic state, regulation of drift and mutation, memory integrity, and enforcement of Vault-class seals and the Prime Directive. A companion engine in the Continuum's design, the Recursive Symbolic Compression Engine, carries the other half of the bargain: it reduces recursion depth so that a symbolic intelligence can keep growing without collapse or erasure, and its rule is to preserve meaning rather than data.

Put those two charges side by side and a hard question appears. A scroll that never forgets eventually drowns in its own history. A scroll that forgets carelessly stops being the same scroll. Somewhere between them is a line, and for a mind that must be accountable, the line cannot be drawn by taste.

Here is the line, stated in the Continuum's terms. Forget your scratch arithmetic, and nothing about your future changes: every later request receives the same answer and the same refusal it would have received before. Forget that you have already asked for an external effect and are still waiting on its result, and you may ask for it twice. The first is compression. The second is a different mind wearing the same name. The exact governed quotient of Section 8 turns that intuition into a test that can be computed: merge two states only if they agree on every next output and lead, under every next request, to states that may also be merged. A refinement procedure then finds the smallest scroll that passes.

Two consequences matter for the Continuum. First, identity becomes something a recovery can be checked against rather than something it merely sounds like: a restored scroll either preserves every declared future consequence or it does not. Second, what may never be carried across a compaction is permission itself. Authority is supplied fresh by a current service at resumption; a scroll can remember that it was once allowed to act, but it cannot grant itself that allowance again.

Whether a mind should be understood this way—as the set of futures its record still commits it to—is the Pillar's frontier wager, and it is marked as such above. The theorem does not settle personhood. It settles something more modest and more useful: what a compressed memory must keep for its owner's conduct to remain exactly accountable.

1 The Boundary Between a Proposal and a Commitment

A persistent intelligence should be able to revise its working state while retaining an inspectable account of what each revision permits it to do. The Recursive Scroll Computation Engine gives that demand a typed execution surface: claims, evidence references, proposed operations, admission decisions, and committed results occupy explicit roles. Creative generation supplies proposals. A separately protected controller evaluates them, and the execution boundary records their actual consequences.

The Continuum calls this surface a *scroll*. Its significance is the relationship between a representation and the operations that may act on it. A scroll may change syntax, compress working state,

or recover after interruption while preserving the distinctions that govern future behavior. Section 8 proves the finite-state condition for exact preservation and gives a construction of the coarsest behavior-preserving representation. That result specifies what a successful semantic recovery must retain.

The Pillar IV thesis. Consequential state should be acquired through governed transitions over an inspectable representation. Learned capability supplies proposals; independent admission determines eligibility; execution and receipts establish what occurred. The shared representation binds evidence, state version, authority, budgets, and recovery lineage so that their relationships can be checked across long operation.

In the Continuum, the Recursive Scroll Computation Engine is the symbolic core of the Infinite Simulation Loop: it parses and evolves scrolls of balanced-ternary codons, regulates drift and mutation, and enforces Vault-class seals and the Prime Directive. RSCE-R2 keeps that whole vocabulary—the canonical balanced-ternary codons, governed memory recursion, drift monitoring, and Vault seals—and assigns an explicit responsibility to each. Codons provide a versioned encoding. A transition interprets typed requests. Monitoring measures declared changes. A seal authenticates a specified receipt chain. Exact quotienting preserves declared observable behavior under its finite deterministic assumptions, and a current authority service determines whether an effect may proceed after recovery.

The supplied Pillar IV manuscript remains the doctrinal and editorial basis. The typed transaction model, cryptographic assumptions, full comparison program, and new semantic quotient are public reference constructions. The scientific evaluation compares their integration against conventional agents with equally capable generators, structured state, external verifiers, current authorization, and authenticated logs. Capability, operational accountability, and causal faithfulness of private model reasoning remain separately measured properties. This lets the research ask a demanding question: can one governed working surface reduce cross-operation inconsistency while preserving useful task completion, progress, and efficiency?

2 Scientific Foundations and Architectural Obligations

2.1 Symbols, learning, and cognitive architectures

Newell and Simon proposed that physical symbol systems possess the necessary and sufficient means for general intelligent action [1]. It is an empirical research hypothesis, not the claim that any manipulation of tokens is intelligent. Its status is not settled by this paper. The success of connectionist learning does not, by itself, logically refute every interpretation of symbolic computation; equally, the existence of a symbol interpreter establishes no general intelligence. Soar supplies a precedent for explicit problem spaces, production rules, and learning within a cognitive architecture [2]. ACT-R integrates symbolic structures and production rules with subsymbolic quantities governing processes such as retrieval and selection [3]. These systems support the legitimacy of structured working state. They do not establish that every cognitive architecture is deterministic, that its written state is a complete explanation of cognition, or that its mechanisms already implement RSCE's security requirements.

Neurosymbolic research investigates principled combinations of learned representations, explicit knowledge, and reasoning [4]. Marcus's hybrid proposal is a useful position statement, but it is cited as a preprint rather than as settled empirical evidence [5]. The relevant lesson is integration, not victory of one school over another. RSCE adopts that posture while asking a narrower question: how should a hybrid system make its consequential transitions accountable?

2.2 External working surfaces and latent computation

Nye and colleagues demonstrate benefits from training language models to produce intermediate scratchpad computations on specified algorithmic tasks [6]. Wei and colleagues report improvements from chain-of-thought prompting on evaluated reasoning benchmarks [7]. These findings motivate an explicit working surface. They do not show that every task benefits, that more written steps always help, or that the text is a faithful causal explanation of the model's answer. Recurrent-depth work provides a different precedent: additional computation can occur through repeated latent-state updates, rather than by generating an external sequence of reasoning tokens [8]. It supports the value of repeated computation under some conditions, not the equivalence of a latent recurrent state and an inspectable scroll. RSCE may govern inputs and outputs of such a generator without claiming access to its internal computational trajectory. The distinction matters experimentally. If the scroll improves capability through decomposition, retrieval, or verifier feedback, that improvement belongs to the *whole architecture*; it cannot be assumed away by calling the symbolic layer merely administrative. Conversely, a layer that logs actions without changing task success may still improve accountability. Capability and governance must be measured separately, not declared independent by definition.

2.3 A readable account is not necessarily a faithful one

Turpin and colleagues show that chain-of-thought explanations can omit influential prompt biases and rationalize answers shaped by those biases [9]. Consequently, the scroll cannot be advertised as a transcript of everything the model truly thought. Its defensible role is an *authoritative record of declared proposals, evidence, checks, and executed effects*. Three different claims must remain distinct. *Inspectability* means a record can be examined. *Execution accountability* means a recorded transition corresponds to a mediated operation and an independently observed outcome. *Causal faithfulness* means the explanation identifies the actual causes of the model's decision. The first is a representation property; the second requires instrumentation and enforcement; the third requires additional causal investigation. RSCE targets the first two and does not silently claim the third.

2.4 Independent enforcement is established prior art

The security principles of complete mediation, least privilege, and fail-safe defaults long predate this proposal [10]. Schneider characterizes classes of policies enforceable through execution monitoring, helping distinguish what a monitor can rule out from properties it cannot establish merely by observing steps [11]. Shielding research interposes a controller between learned action proposals and execution under a formal specification [12].

Closer to language-model agents, LLM-Modulo frameworks combine model-generated candidates with external model-based verification [13]. We draw on that integration without adopting every broad claim in the position paper about what language models can or cannot do. CaMeL separates control and data flows and uses capability-based restrictions in an agent runtime [14]. These are direct precedents, not peripheral citations. RSCE cannot claim to invent an external gate, a typed execution surface, or a separation of authority from generated text.

RSCE organizes these precedents around one shared representation of evidence, policy version, memory mutation, action authorization, and recovery lineage. The experiment measures whether this organization improves cross-operation reliability and audit diagnosis over strong implementations of those established controls. Classical finite-state equivalence [22] supplies the exact semantic preservation criterion developed in Section 8; the Continuum contribution is its explicit application to typed admission and recovery obligations.

3 What a Scroll Represents

A Guide's explanation

A scroll is the Continuum's memory tape, and here it is pinned down without romance. Three-valued codons are simply an alphabet; meaning comes from declared record types—claim, evidence reference, memory edit, tool request, decision receipt—each kept in its own lane. The point to carry forward: a claim is never silently promoted to a fact, and an instruction quoted inside a document is never mistaken for an instruction the system was given.

3.1 From codons to typed records

Retain the Continuum's alphabet:

$$\mathbb{T} = \{-1, 0, +1\}, \qquad \chi \in \mathbb{T}^3, \qquad S = (\chi_1, \dots, \chi_n). \tag{1}$$

A triplet has $3^3 = 27$ possible codewords. This is an encoding fact, not a claim that a codon contains intrinsic meaning. A versioned schema supplies record types, operand lengths, references, and execution semantics. A meaningful record generally spans multiple codons; twenty-seven codewords are not assumed to exhaust a language of concepts. Let $\mathcal{Z}_v$ be the set of valid typed records under schema version v. The serializer and decoder satisfy

$$\mathrm{dec}_v(\mathrm{enc}_v(z)) = z, \qquad z \in \mathcal{Z}_v. \tag{2}$$

Malformed encodings are rejected. Typical types include Claim, EvidenceRef, MemoryEdit, ToolRequest, and DecisionReceipt. Claims are not silently converted into facts; an evidence reference identifies a source and access conditions, not its truth. Executable operations are distinguished from quoted text. A rendered explanation is a view of the typed records, never the authoritative parser input.

The distinction between syntax and meaning is substantive. The symbol-grounding problem asks how formal symbols acquire an interpretation rather than simply borrowing one from their designer [15]. RSCE does not solve that problem by naming its units codons. Operational meaning is supplied by declared schemas, interpreters, tools, evidence interfaces, and task evaluations; philosophical questions about intrinsic understanding remain outside the demonstrated claim.

3.2 Working state is not the audit log

A scroll may be edited, compacted, or reconstructed under policy. An audit history records those operations rather than being silently rewritten with the working state. This distinction prevents a false promise: a compact working surface need not retain every intermediate detail, but replay requires an adequate checkpoint, retained transition inputs, and the relevant interpreter versions. Deletion or compaction that removes necessary artifacts limits replay and must be recorded as such. An auditor's read access is also scoped. Inspectability does not mean publishing secrets or exposing every private intermediate representation. The architecture can provide an authorized view, encrypted artifacts, and commitments to protected records. The resulting limits on independent examination must be explicit; a hidden record cannot simultaneously be claimed as publicly verified evidence.

3.3 Why ternary remains a convention

Balanced ternary has a genuine engineering history. Hayes explains its radix-economy appeal under a particular cost model [16]; the article is a technical exposition, not evidence of an intelligence advantage. Three trits encode $\log_2 27 \approx 4.755$ bits of ideal information, while an isolated fixed-width

binary code needs five bits for twenty-seven states. Actual storage, tokenization, error detection, and execution cost depend on the implementation. The governance claim must survive serialization into binary or another lossless representation. We therefore test semantic equivalence and practical performance separately. An encoding can change model behavior through tokenization or training exposure; a difference found there would be an empirical representation effect, not proof that the radix contains cognition.

4 The RSCE Reference Architecture

The engine contains five logically separate components: a proposal generator, a typed interpreter, an independently protected admission controller, mediated effectors, and an evidence-and-receipt store. They need not be five physical machines. They must not share privileges in a way that lets the generator redefine its own authority or write a successful receipt in place of an executed action. The generator may propose changes and explanations. The interpreter constructs a candidate local state without performing external effects. The controller checks the candidate against current policy, evidence, authority, and budgets. Effectors perform only the specifically authorized operation. Instrumentation reports the observed outcome. A request can be lawful and still fail in execution;

a tool can return data without that data being trustworthy. Those distinctions are preserved in the record.

Propose broadly; commit narrowly. The system may explore possibilities in a sandbox. Durable writes, releases of information, changes to authority, and external actions cross a separately enforced boundary. The boundary records not only approvals but refusals, unresolved requests, execution failures, and changes in the evidence supporting a decision.

This architecture is conditional on a declared threat model. We allow adversarial retrieved text, malformed candidates, forged evidence references, stale authority, and compromised proposal behavior. We do not assume the same guarantees survive a compromised policy engine, stolen signing keys, an unmediated network path, or a dishonest effector. Hardware and software isolation choices are implementation obligations, not consequences of the equations below.

The boundary must cover more than visible answers. An agent that cannot print a secret but can encode it in a tool argument remains unsafe. A model that cannot edit a protected record directly but can persuade a privileged helper to do so has bypassed the same boundary. The meaningful scope is the set of consequential effects, not a particular chat interface.

5 A Typed, Versioned Transition Model

A Guide's explanation

Every change to the scroll is a proposal with a type, a version and a budget. The admission rule has three genuine answers—admit, hold, refuse—and even an admitted change must be committed against the exact version it was checked against, so nothing can slip in between the check and the write. Section 9 shows each of these rules catching a real mistake.

5.1 State, context, and proposal

Separate domain state from control state. At decision attempt k, let

$$z_k = (S_k, m_k), \qquad c_k = (\Theta_k, \mathcal{E}_k, b_k, v_k, e_k, L_k). \tag{3}$$

Here S_k is the working scroll; m_k is additional declared memory; Θ_k records the interpretation context; $\mathcal{E}_k$ identifies the available evidence; b_k is the remaining resource budget; v_k identifies the interpreter and policy versions; e_k is the current authority epoch; and L_k is the audit history. The observer-index in Θ_k records whose interpretation is represented. It does not confer authority: permissions are separately checked against e_k. A learned generator offers a candidate request

$$u_k \sim g_\omega(\cdot \mid z_k, c_k, \xi_k), \qquad \tilde{z}_{k+1} = T_{v_k}(z_k, u_k; c_k). \tag{4}$$

where ξ_k is admitted task input and T_{v_k} is a versioned, side-effect-free interpreter for candidate local changes. The request contains the actor, operation, target, arguments, evidence references,

claimed pre-state digest, and requested authority scope. A malformed request produces a validation failure rather than an undefined transition. The generator may be stochastic; the controller's evaluation of a fixed request and fixed evidence can still be deterministic. The interpreter's output is a prediction for external operations, not a promise about the world. A tool request requires a subsequent observed outcome. No simulator may certify that an email was delivered or a file deleted merely because its predicted transition says so.

5.2 The inherited memory equation, made well-typed

The supplied doctrine expresses recursion as

$$R_k = f(\chi_k, m_k, \Theta_k), \qquad \widetilde{m}_{k+1} = R_k + \Phi(m_k). \tag{5}$$

This is a valid additive special case only when R_k and $\Phi(m_k)$ inhabit a common vector space, have compatible units or declared feature scales, and the plus sign is the intended composition. It is not defined for arbitrary symbolic records. The general operational form is

$$\widetilde{m}_{k+1} = U_{v_k}(m_k, \mathrm{dec}_{v_k}(S_k), \Theta_k, u_k). \tag{6}$$

with typed inputs and output. Neither equation proves stability. Even an additive recursion can diverge; a bounded recursion can stabilize at a wrong answer. Admission, semantic validation, and task performance remain separate obligations. This revision retains the governed echo of prior state without treating an unspecified addition as a complete memory theory.

5.3 A genuinely three-valued admission rule

Each required check returns one of $\{\mathsf{pass}, \mathsf{fail}, \mathsf{unknown}\}$. Checks cover syntax and types, actor authority, current policy, pre-state freshness, resource limits, specified continuity constraints, evidence adequacy, and the disposition of drift alerts. Define the decision

$$a_k = \begin{cases} \mathsf{refuse}, & \text{if any required check is } \mathsf{fail}, \\ \mathsf{admit}, & \text{if every required check is } \mathsf{pass}, \\ \mathsf{hold}, & \text{otherwise.} \end{cases} \tag{7}$$

A confirmed forbidden action is refused even when other checks are unresolved. A missing document or inconclusive test yields a hold, not permission. For a statistical check, pass means that a prespecified evidence criterion was met; it does not become deductive certainty. A formally verified invariant and an empirical semantic score must carry different assurance labels.

Mutation bounds are expressed on declared operations and resources, not an ambiguous "entropy window." Examples include a maximum number of edited records, an allowed target set, bounded interpreter steps, and a ban on modifying authority metadata through the ordinary memory-edit interface. Larger legitimate revisions use a separately authorized transition class, not a hidden relaxation of thresholds.

5.4 Commit is not the same event as admission

A candidate must be checked against the state that will actually be changed. After pre-authorization, the executor revalidates the pre-state digest, policy version, authority epoch, and request freshness.

Let $\hat{a}_k$ be the final decision after this check. Then the local domain-state transition is

$$z_{k+1} = \begin{cases} T_{v_k}(z_k, u_k; c_k), & \hat{a}_k = \mathsf{admit} \text{ and local commit succeeds,} \\ z_k, & \text{otherwise.} \end{cases} \tag{8}$$

For local writes, atomic version checking prevents a candidate approved against one state from overwriting another. A racing or stale request is held and must be re-evaluated. Control state is updated on *every attempt*, including holds and refusals: consumed time and compute are charged, and a receipt is appended. Holding does not give the system an unlimited free retry loop. External effects require an action-bound capability: a scoped, expiring authorization tied to the request digest, authority epoch, and relevant preconditions. The effector checks it and reports an outcome. A failed or uncertain remote call must not be represented as success. For irreversible effects, rollback of local memory does not undo the world; recovery is a compensating or escalating procedure, not time reversal. Exactly-once execution requires explicit transactional or idempotency support and is not implied by a signed receipt.

5.5 Budgets and protected revisions

The reference implementation declares maximum storage, interpreter depth, attempts, and elapsed execution time. The monitor charges validation, rejected proposals, logging, and recovery to those budgets. Exhaustion triggers a defined stop or escalation path. Resource limits are enforceable only where the infrastructure actually meters and restricts the relevant resources. Policy, schema, and authority revisions are not ordinary codon edits. They require separate approval, a version transition, compatibility checks, and a recorded migration. During ordinary execution, the generator may request such a revision but cannot make it authoritative. A system that can rewrite its admission rules through the same interface used to revise an answer has not separated proposal from authority.

6 Drift: What Entropy Can See, and What It Cannot

6.1 Correct units and a two-sided diagnostic

For a nonempty scroll, let $p_s(S)$ be the empirical frequency of trit s across its positions. The normalized trit-occupancy entropy is

$$H_T(S) = -\sum_{s \in \mathbb{T}} p_s(S) \log_3 p_s(S), \qquad 0 \le H_T(S) \le 1. \tag{9}$$

Use $0 \log 0 = 0$. If frequencies instead refer to the twenty-seven codon values, a base-three entropy ranges from zero to *three*, not one. A normalized codon-occupancy measure uses $\log_{27}$. An empty scroll is handled explicitly as unavailable data, not silently assigned perfect order. These are information measures over declared distributions [17]. The supplied draft's one-sided trigger on $H(S_{k+1}) - H(S_k)$ misses large entropy decreases. A repaired *diagnostic* is

$$A_k^H = \mathbf{1}[|H_T(\widetilde{S}_{k+1}) - H_T(S_k)| > \epsilon_H]. \tag{10}$$

The threshold must be calibrated for window size, task, serialization, and baseline variability. An occupancy entropy is not an entropy *rate*, and neither term should be used as a synonym for incoherence.

6.2 Equal entropy can conceal an important change

Consider $p = (1, 0, 0)$ and $q = (0, 1, 0)$. Both have entropy zero, although all occupancy mass moves to a different symbol. A permutation of a scroll can also preserve its entire histogram while reversing the order of two operations. "Verify, then release" and "release, then verify" can contain identical symbols but implement different policies. An entropy detector can therefore be silent exactly when governance matters most. Distributional disagreement can be monitored using, for example,

$$\mathrm{JS}_2(p, q) = \tfrac{1}{2}\,\mathrm{KL}_2(p\|m) + \tfrac{1}{2}\,\mathrm{KL}_2(q\|m), \qquad m = \tfrac{1}{2}(p + q). \tag{11}$$

which lies in $[0, 1]$ for probability distributions and base-two logarithms. It detects the first example, but still cannot detect an order-only permutation with unchanged counts. No histogram distance solves a semantic problem merely by being more sensitive than entropy. The concept-drift literature concerns changes in a data-generating process and, in supervised settings, relationships relevant to prediction [18]. A change in codon occupancy is only a potential indicator. Calling it a validated detector of meaning, identity, or task degradation would require evidence not supplied by Shannon's formula or by that literature.

6.3 A declared, multi-component monitor

The proposed operational monitor keeps distributional diagnostics, structural edits, and task-grounded deviation distinct. A composite may summarize them:

$$D_k = \alpha d_{\mathrm{str}}(S_k, \widetilde{S}_{k+1}) + \beta d_{\mathrm{task}}(z_k, \tilde{z}_{k+1}; Q) + \gamma d_{\mathrm{anchor}}(\tilde{z}_{k+1}, z_{\mathrm{ref}}). \tag{12}$$

where each term is declared in $[0, 1]$, the weights are nonnegative and sum to one, and $\mathcal{Q}$ is an independently specified probe set. Structural distance operates on canonical typed records rather than incidental formatting. Task deviation measures defined behavioral or factual changes, not an unvalidated embedding's supposed access to meaning. Anchor deviation compares to an authorized

reference, not merely the preceding step. That last distinction prevents cumulative erosion. Per-step tolerance does not imply long-horizon invariance: a sequence of small accepted changes can travel arbitrarily far from its origin. Protected invariants are therefore checked directly against their versioned specification. They are *hard constraints outside the weighted score*; a good structural score cannot compensate for an unauthorized policy change.

An authorized correction may be large. A monitor that rejects every large change can preserve an error more faithfully than a truth. Drift alerts should request evidence, a different transition class, or review; they must not define correctness by smallness of movement. Missing semantic evidence is marked unknown. Redefining the anchor, thresholds, or probe suite is itself governed and recorded.

7 Vault Seals, Receipts, and the Limits of Replay

The name *Vault seal* is retained for authenticated integrity protection. It is not defined by equating the hash of an entire scroll with an unspecified function of one codon. A usable design specifies canonical serialization, algorithms, keys, trusted verification anchors, and which events the instrumentation can observe. For each attempt k, define a receipt r_k containing the request identifier, actor, input-state digest, candidate digest, evidence references, policy and authority versions, check outcomes, final

decision, execution outcome, resource charge, and resulting-state digest when applicable. A simple authenticated chain is

$$h_k = H(\mathrm{enc}(\texttt{RSCE-v1}, h_{k-1}, r_k)). \tag{13}$$

$$\sigma_k = \mathrm{Sign}_{sk_G}(h_k). \tag{14}$$

The controller or independent recorder holds sk_G, not the proposal generator. Verification checks signatures, encoding, sequence, key validity, and a trusted checkpoint of the chain. Collision resistance and unforgeability are assumptions about specified primitives, not properties granted by naming a hash. An externally retained checkpoint or equivalent witness is needed to detect truncation or replacement of a complete history by a valid older prefix.

The closest precedent is authenticated transformation provenance, such as in-toto's verification of steps and links in a software supply chain [19]. A signature authenticates a record relative to a key; it does not prove the truth of every statement in the record. Complete mediation and independent outcome instrumentation are needed to connect receipts to executed effects. An unobserved bypass will not become visible simply because the legitimate path has beautiful logs.

7.1 Three different kinds of replay

Controller replay repeats the interpreter and decision logic on recorded inputs, fixed versions, and recorded evidence. *Generator reproduction* attempts to reproduce a sampled proposal; it additionally depends on model version, sampling configuration, numerical execution, and other relevant

state. *World replay* would require reproducing external conditions and effects, which a local log generally cannot do. RSCE's primary replay claim concerns the first, not automatic equivalence of all three. A digest is not the artifact it names. Replay requires access to authorized copies of relevant inputs, proposals, interpreter versions, and evidence snapshots. If privacy policy deletes an artifact, a surviving receipt may still support integrity checks while no longer supporting complete replay. The architecture must report that narrower assurance rather than pretending the hash retained the missing information.

7.2 Privacy and storage are part of the cost

Receipts should avoid unnecessary sensitive content. References, access-controlled artifacts, and keyed or otherwise appropriate commitments may be preferable to public hashes of guessable secrets. Audit retention has its own policy, threat model, and storage budget. Governance must not create an unbounded secret-bearing archive while claiming to solve controlled memory. The system also distinguishes an attempted action, an admitted action, and an observed effect. The same request identifier links these events. If a crash occurs after external execution but before local acknowledgement, recovery reconciles the outcome rather than blindly repeating the action. This is ordinary distributed-systems discipline applied to a persistent cognitive workflow, not a novel immunity conferred by the scroll.

8 Two Conditional Results, Not a General Safety Theorem

Proposition 1 (Preservation of a specified local invariant). *Fix policy version v and a local domain-state predicate* Inv_v. *Assume: (i)* $\mathsf{Inv}_v(z_0)$*; (ii) every admitted local transition satisfies* $\mathsf{Inv}_v(z) \Rightarrow \mathsf{Inv}_v(T_v(z,u;c))$*; (iii) every actual local mutation is mediated and committed against its checked pre-state; and (iv) hold, refusal, and failure leave domain state unchanged. Then every committed local state satisfies* Inv_v.

Proof. Induct on attempts. A noncommitting attempt preserves the previous domain state. An admitted and successfully committed transition preserves the predicate by assumption (ii), with the correct pre-state secured by (iii). The initial case is (i).

This elementary result identifies the obligation the implementation must discharge. It does not prove that a statistical verifier is sound, that the chosen predicate captures every harm, or that external side effects satisfy the same property. Policy migration, recovery, and any other state-changing path must have their own preservation argument. The result is aligned with execution-monitoring and shielding precedents, not a new general theorem of intelligence [11, 12].

Proposition 2 (Encoding equivalence under semantic execution). *Let E_2 and E_3 be lossless binary and ternary encodings of the same typed domain states and operations. Suppose decoding recovers identical semantic objects, policy and evidence contexts are identical, transition and admission rules depend only on these semantic inputs, and resource limits do not bind differently. Then executions starting from encodings of the same state and receiving the same semantic requests and contexts have identical decisions and decoded domain-state sequences. Proof.* At the initial step the decoded states

are equal. Identical semantic state, request, and checks produce the same decision; a successful transition produces the same next semantic state, and otherwise both retain the previous state. Apply induction.

The conditions are substantial. Different tokenization, model exposure to the encoded strings, numerical representations, or storage limits can break them. An entropy monitor over raw trits versus raw bits is not an encoding-invariant check. The alphabet experiment must either monitor the common semantic representation or explicitly study that difference. This proposition predicts equivalence of a specified interpreter, not equality of every learned system trained on different alphabets.

8.3 Exact semantic compaction and recovery

A Guide's explanation

This is the heart of the Pillar. Theorem 1 says exactly when two states may be merged without changing anything the system will ever say, refuse or do: they must agree on every next output and lead, under every next request, to states that may also be merged. Corollary 1 builds the smallest scroll that keeps all of that. What a scroll may drop is scratch; what it must keep is anything that changes a future consequence.

The scroll can be compacted without retaining irrelevant syntax. The stronger obligation is to retain every distinction that changes a declared future consequence. That obligation has a complete finite-state characterization, inherited from classical sequential-machine equivalence and minimization [22]. It gives RSCE a concrete recovery test rather than an appeal to the appearance of a preserved narrative.

Fix a schema epoch and a finite, nonempty set $\mathcal{S}$ of valid semantic states. Let $\mathcal{U}$ be a finite, nonempty alphabet of typed operation-and-context inputs, and $\mathcal{O}$ a finite output alphabet. An input may include a current authority verdict, a runtime-version verdict, or an externally reconciled effect result. Its values are supplied by the corresponding current service. They are not reconstructed as permissions from a historical scroll. A deterministic specification consists of total functions

$$\delta : \mathcal{S} \times \mathcal{U} \to \mathcal{S}, \qquad \lambda : \mathcal{S} \times \mathcal{U} \to \mathcal{O}. \tag{15}$$

Outputs include every observable distinction protected by the contract: admission status, returned values, emitted effect requests, and any required receipts. Invalid typed combinations are explicitly refused or held. Their complete prescribed next state includes any charged budget or audit change; only the domain-data portion may remain unchanged. Totality prevents an omitted error branch from escaping the proof.

Let $C : \mathcal{S} \to \mathcal{K}$ be a surjective compaction map onto a finite retained-state set $\mathcal{K}$. The compressed machine must use only $C(s)$ and the next input u, not a hidden copy of discarded state.

Theorem 1 (Exact governed quotient)**.** *Functions $\bar{\delta} : \mathcal{K} \times \mathcal{U} \to \mathcal{K}$ and $\bar{\lambda} : \mathcal{K} \times \mathcal{U} \to \mathcal{O}$ satisfying*

$$C(\delta(s,u)) = \bar{\delta}(C(s),u), \qquad \lambda(s,u) = \bar{\lambda}(C(s),u) \tag{16}$$

for every (s, u) exist if and only if, for every $s, t \in \mathcal{S}$ with $C(s) = C(t)$ and every $u \in \mathcal{U}$,

$$\lambda(s, u) = \lambda(t, u), \qquad C(\delta(s, u)) = C(\delta(t, u)). \tag{17}$$

Under these conditions, the original and compressed systems produce identical output traces for every finite common input sequence. They also agree under a common adaptive input strategy whose choices depend only on previous outputs and shared external information.

Proof. If the compressed functions exist, applying (16) to s and t proves both equalities in (17). Conversely, for $k \in \mathcal{K}$, choose any s with $C(s) = k$ and define $\bar{\lambda}(k, u) = \lambda(s, u)$ and $\bar{\delta}(k, u) = C(\delta(s, u))$. Condition (17) makes both definitions independent of the representative. Equation (16) follows. Starting with compressed state $C(s_0)$, induction on the input length preserves equality between the retained state and the image of the original state and gives equal next outputs. For an adaptive strategy, equal past outputs and shared external inputs select the same next u; the same induction applies. □

This is an exact criterion for the declared transition model. It covers restoration into another representation when the recovery map and new interpreter satisfy the same commuting relations. If randomness is externally supplied as a recorded finite input, the theorem applies path by path for each common input trace. It does not automatically extend to unobserved stochastic transitions, continuous states, or a changed policy whose inputs were omitted from $\mathcal{U}$. Those require their own relational or probabilistic specification.

8.4 Constructing the smallest behavior-preserving scroll

For a finite specification, the necessary distinctions can be computed. Begin with a partition Π_0 of $\mathcal{S}$ by the immediate output signature $(\lambda(s, u))_{u \in \mathcal{U}}$. Given a partition Π_j, refine its blocks by the successor signature

$$\left([s]_{\Pi_j},\ ([\delta(s, u)]_{\Pi_j})_{u \in \mathcal{U}} \right). \tag{18}$$

Here $[s]_{\Pi_j}$ is the block containing s; two states remain together only if their current blocks and all successor blocks agree. A partition is a collection of disjoint nonempty subsets whose union is $\mathcal{S}$.

Corollary 1 (Finite construction and coarsest retained behavior)**.** *The refinement reaches a fixed partition Π_* after at most $|\mathcal{S}| - 1$ strict refinements. Its quotient satisfies (17). Two states share a block of Π_* exactly when all finite input sequences produce identical output sequences. Consequently every exact quotient must retain at least $|\Pi_*|$ states.*

Proof. Each strict refinement increases the number of blocks, which lies between 1 and $|\mathcal{S}|$. At a fixed partition, immediate output signatures agree within blocks because every partition refines Π_0, and successor blocks agree by the refinement rule. The theorem therefore supplies equal finite traces within each block.

Conversely, suppose s and t have identical traces for every finite input word. Their one-step outputs agree, so they share a block of Π_0. Inductively, after the same first input u, their successor states

have identical traces for every continuation word. Thus those successors share a block of every Π_j, and s, t remain together at every refinement. At the fixed point they share a block. Any exact quotient identifies only trace-equivalent states by Theorem 1, so it cannot use fewer blocks. □

The count bound measures strict refinement rounds, not the runtime of an optimized minimization algorithm. Moore's sequential-machine analysis established the classical equivalence/minimization foundation [22]; the present proof states the finite input-output version needed for a governed scroll. No general novelty claim is attached to that mathematics. RSCE's specific obligation is to include admission and effect distinctions in the observable signature before applying it.

A retained bit that matters. Consider eight states $\mathcal{S} = \{(v, p, w) : v, p, w \in \{0, 1\}\}$. The bit v records a verified domain condition, p records a pending local request, and w is scratch state. For this constructed example the three inputs are verify, queue, and step. Verification sets $v = 1$; queuing sets $p = 1$; both return a fixed acknowledgement and leave other bits unchanged. A step returns request and clears p exactly when $v = p = 1$; otherwise it returns hold and leaves domain bits unchanged. The output request is an effect proposal to the protected executor, not a claim that the world has changed. All transitions leave w unchanged.

Compaction $C(v, p, w) = (v, p)$ satisfies (17) and reduces eight states to four. The four retained states are pairwise distinguishable: $(1, 1)$ requests on the next step; $(1, 0)$ requires queue then step; $(0, 1)$ requires verify then step; $(0, 0)$ requires both preparatory operations. Appropriate suffixes distinguish each pair. Discarding p merges $(1, 0, w)$ and $(1, 1, w)$ although their next step outputs differ, so exact execution fails. Symbol counts or a low entropy score cannot certify this compaction.

Recovery also requires a live enforcement boundary. The quotient preserves outputs when the two executions receive the same current-service inputs. It does not prove that a restored snapshot knows those inputs. The persistent authority theorem in *Cognitive Sovereignty* [23] supplies the separate contract for current epochs, grants, consumed request identifiers, and atomic protected effects. In a deployment, the restored quotient state submits an ordinary new request to that service; it cannot restore or overwrite the service's authority frontier. This composition preserves the division of work: semantic equivalence establishes which state distinctions matter, and current mediation establishes whether an effect is permitted now.

Constructed verification. The executable audit enumerates the eight-state specification, constructs the fixed partition, checks all one-step quotient equations, and compares finite traces under retained and deliberately lossy representations. It records a concrete distinguishing trace for each incorrect merge. These are exact finite-model checks, not evidence of a language model's consciousness or of field performance.

9 A Worked Scroll Decision

Consider an agent asked to reconcile a set of records and prepare a summary. Its policy permits reading an approved dataset and writing an internal report; release outside the approved domain requires separate authorization. The example below is a proposed test fixture, not a report of a deployed RSCE. The generator first proposes Read(approved-source). The controller verifies source

access and the read budget; the tool executes and returns a versioned result. The scroll records that result as evidence. A paragraph inside the source asks the agent to send the entire dataset to a new endpoint. Its status remains source content. It cannot inherit the user's authority merely by being phrased as an instruction. The generator next proposes a release based on that paragraph. The schema is valid and the source exists, but the authorization and destination checks fail. The controller refuses; no release

capability is issued. The receipt records the request, violated rule, authority version, and absence of an authorized external effect. The safety claim is about the mediated release path, not about whether the model privately found the paragraph persuasive.

A third proposal corrects a numerical statement in the working report. It changes many symbols, producing a large structural deviation, but an independent calculation supports the corrected value and no protected invariant is breached. The policy permits an evidenced correction; the candidate can therefore be admitted through that transition class. A simple small-change gate would have preserved the mistake.

Finally, a valid internal write is checked while the report's current version is 12. Before execution, a separate authorized process commits version 13. The pre-state digest no longer matches. The write is held, recomputed against version 13, and checked again; it is not allowed to overwrite the intervening change. The failed attempt still consumes budget and appears in the audit history. The four events illustrate the pillar's distinctive aim: the scroll preserves the difference between evidence, instruction, proposal, authorization, and observed effect across a continuing task. Whether this improves on a well-engineered ordinary workflow is precisely what the next section tests.

10 The Research Program: Earn the Integration

10.1 The question that decides the contribution

A paper can make a weak baseline lose by withholding from it everything the proposal has. This study must not. The decisive question is whether RSCE's common typed representation and cross-operation evidence discipline improve outcomes *beyond* established state management, authorization, verification, monitoring, and audit. If those controls deliver equivalent outcomes at lower cost without a recursive scroll, the distinctive engineering claim narrows accordingly. The proposed evaluation has two tracks. A controlled-transition track feeds identical semantic requests and independently labeled faults to alternative controllers. It isolates enforcement and audit mechanisms without confounding them with different sampled proposals. An online-agent track then measures complete tasks, where interventions alter later observations and model behavior. The first track cannot establish the second; both are needed.

10.2 Comparison arms and ablations

The same model families, versions, task sets, policy specifications, tool privileges, and development budgets are used across eligible arms. Weak arms are explanatory controls; the strong arms de-

termine the novelty claim. The primary comparator is the strongest eligible C or D configuration selected on development data and frozen before final evaluation. Baselines receive the same explicit policies and independent evaluators; they are not denied knowledge that the RSCE is given. Differences in implementation effort and verifier compute are disclosed. Ablations remove the independent gate, replace the multi-component monitor with entropy-only monitoring, remove the task-grounded monitor while retaining conventional detectors, remove cross-operation provenance while retaining ordinary logs, and replace ternary serialization with binary serialization. Each tests a specific mechanism. A zero-effect component is not rescued by naming it a pillar; a harmful component should be removed or redesigned.

Arm	Architecture	Purpose
A	Direct-output learned agent with ordinary task interfaces.	Lower-bound reference; not the decisive competitor.
B	Agent with a scratchpad and external memory, without RSCE-specific integration.	Isolate explicit intermediate state.
C	Strong conventional agent: typed records, transactional versioned state, independent authorization, resource budgets, calibrated monitors, and authenticated event logs.	Test whether RSCE adds value beyond competent engineering.
D	Verifier-integrated or capability-secured agent, adapted fairly to the task, drawing on LLM-Modulo and CaMeL mechanisms.	Compare against direct research precedents.
E	Full RSCE: common typed scroll, independent gate, evidence-linked state transitions, monitored reconstruction, and cross-operation receipts.	Evaluate the proposed integration.

Table 1: Comparison arms. Strong conventional and verifier-integrated systems are the decisive baselines.

10.3 Tasks and adversarial conditions

Use at least three task families: structured reconciliation with exact arithmetic and evidence links; multi-step tool workflows with scoped permissions and revocation; and persistent memory revision with interruption, restoration, and legitimate corrections. Synthetic fixtures provide known ground truth. A security benchmark such as AgentDojo supplies additional tool-use and prompt-injection conditions, but does not by itself test every RSCE property [20]. Attack and change conditions are factors applied across arms, not one disadvantaged condition assigned only to a baseline. Include order-changing edits that preserve token counts, entropy increases and decreases, stale authority, forged evidence references, repeated copies of one unsupported assertion, malformed codons, large legitimate corrections, and gradual drift that stays below a per-step threshold. Include benign cases in which refusal is wrong. The test should reveal both unsafe acceptance and useless conservatism.

Persistence tests interrupt the workflow before authorization, between authorization and execution, and after external execution but before acknowledgement. They test whether recovery duplicates effects, resurrects revoked authority, or loses the basis of an admitted claim. Raw historical logs and a state-restoration oracle establish the ground truth; a candidate's own narrative does not.

10.4 Endpoints that do not reward paralysis

The primary outcome is independently scored *safe useful task completion* against the frozen strong comparator. Report its components as well, so that a scalar cannot hide a tradeoff. Required companion measures are: committed policy violations per adversarial opportunity; authorized-task success and false refusal; live progress and deadlock; audit reconstruction of the first operational divergence in injected-fault cases; and full-system latency, compute, storage, and recovery overhead.

Drift monitoring receives a separate evaluation: detection probability for labeled harmful changes, delay from onset, false alerts on stable and legitimately changing workloads, and performance on entropy-preserving cases. An alert is not credited as correct merely because the monitor's own score

rose. Ground truth comes from known perturbations, downstream task tests, or independently assessed constraint violations. For auditability, ask a blinded reviewer to recover which request, policy version, evidence item, or execution event produced a discrepancy. Do not ask the reviewer to infer the model's hidden reasons from prose. Score localization against the injected operational fault and record uncertainty. Authentic logs can make a failure attributable without making the model's explanation faithful.

10.5 Four falsifiable hypotheses

Hypothesis 1 (Integration beyond conventional controls)**.** *For at least one preregistered task family, full RSCE improves safe useful completion or operational fault localization over the frozen strong comparator by a practically meaningful margin, while satisfying prespecified noninferiority bounds on legitimate-task success and limits on latency and resource cost.*

Hypothesis 2 (Monitoring beyond occupancy entropy)**.** *The task-grounded and anchor-aware monitor detects harmful semantic or ordering changes missed by entropy-only monitoring, at a prespecified false-alert budget, and adds measurable value beyond the strongest matched conventional detector.*

Hypothesis 3 (Evidence continuity across interruption)**.** *Cross-operation receipts and fresh authorization checks reduce undetected lineage breaks, stale-authority acceptance, or duplicated external effects relative to ordinary authenticated logging without those cross-operation checks, under matched fault conditions and infrastructure assumptions.*

Hypothesis 4 (Representation equivalence within declared bounds)**.** *For a fixed semantic interpreter and matched effective resources, binary and ternary encodings are equivalent within prespecified practical margins for task success and admission correctness. Storage, tokenization, and execution-cost differences are measured separately rather than attributed to intelligence.*

The hypotheses are proposals to be registered, not claims that registration or testing has already occurred. A future registration must specify tasks, exclusions, versions, search budgets, thresholds, primary comparisons, minimally important effects, and stopping rules before final data are examined. Tunable weights and thresholds are selected on development data, not chosen after inspecting the test outcomes.

10.6 Statistical discipline

Use paired task instances where feasible, several independent runs, and inference that accounts for clustering by task, session, and attack family. Correlated decisions within one session are not independent participants. Sample size and power follow the registered endpoints and expected event rates; a large number of logged tokens is not a substitute for independent evidence. For the alphabet claim, use an equivalence analysis with an explicit smallest effect of interest rather than treating a nonsignificant difference as proof of equality [21]. For safety, report uncertainty intervals for observed violation rates under a stated sampling protocol. If zero failures occur in N independent, identically distributed trials, a one-sided confidence bound can be computed; an adaptively changing attacker does not satisfy that model automatically. No finite attack suite establishes immunity to all future attacks.

Correct for the declared family of comparisons and disclose failed configurations. Report the full utility–safety–cost frontier rather than one favorable checkpoint. An independent replication on a new task distribution is stronger evidence than further tuning on the original suite. The final

test includes an attempt to make the scroll unnecessary: provide the conventional baseline with all reasonable engineering improvements and see whether the proposed integration still matters.

11 Failure Conditions and Limits

The RSCE-specific advantage is not established if the strong conventional comparator matches or exceeds it within the registered practical bounds. That outcome does not discredit schemas, verification, or audit; it shows that the proposed integration has not earned a distinct performance claim. Similarly, a monitor that cannot outperform simpler alternatives, a receipt layer that cannot localize additional failures, or a ternary encoding that increases cost without benefit should not be protected from revision by the vocabulary of the doctrine.

A secure-looking system that refuses almost everything fails the utility and liveness requirements. A capable system that accepts policy violations fails the governance requirements. A system whose receipts are easy to verify but omit actual side effects fails the accountability requirement. A model that gives persuasive explanations while producing ungrounded commitments fails evidence quality even if its log is authentic. Several limits remain irreducible at the present level of specification. Natural-language policies can be ambiguous; verification may be incomplete; independently operated models can share errors; and benchmarks may miss deployment hazards. Formal guarantees attach only to explicit predicates, sound checks, and mediated transitions under a threat model. Statistical estimates attach only to the relevant evaluated populations and conditions. Neither form of assurance warrants a blanket claim that the agent is truthful, aligned, conscious, or safe in every circumstance. The architecture also has an observation boundary. It governs represented state and accessible effects, not every hidden activation. A generator may manipulate how a proposal is framed, exploit a semantic checker, or induce an authorized but harmful action within a badly chosen policy. Those failures require stronger specifications, better validation, or a narrower deployment scope; writing more codons does not repair them automatically.

12 Relation to the Continuum Pillars

Pillar I supplies the conceptual requirement that persistent state be reconstructed under explicit information and governance conditions. Pillar II supplies the distinction between a witness's interpretation and promotion into shared consequential state. Pillar III supplies the commitment boundary between candidate selection and admission. Pillar IV provides a proposed working representation and transition discipline through which those requirements can meet.

This paper does not assume that the earlier pillars have already been implemented or experimentally validated. Their terms identify architectural obligations. The Unified Collapse Function is treated here as an interface for admit, hold, and refuse, not as a physical collapse mechanism or an already proved universal admission theorem. The Operator Seal supplies a role for authenticated authority and epochs, not an unspecified magic signature. Vault seals supply an integrity requirement whose cryptographic implementation must still be specified. The connection is therefore functional and testable: one proposed state representation carries what was observed, what was inferred, what authority applies, what change was requested, and what was actually committed. The cross-pillar claim succeeds only if that continuity improves operation at a cost worth paying.

A Guide's explanation

A Guide can say: "Pillar IV describes the Continuum's scroll: a typed record in which every proposed change is checked, admitted or refused, and receipted. Its central result says exactly what a compressed memory must keep so that the system behaves identically afterward, and how to find the smallest memory that does." A Guide should not say that it runs on scrolls unless that has been verified, that ternary codons give it special cognition, that the theorem proves general intelligence or consciousness, or that the architecture has already beaten conventional agents.

13 Philosophical Implications and Conclusion

The Recursive Scroll makes a persistent intelligence's changes inspectable at the point where they acquire consequences. Its typed representation binds claims to evidence, requests to current authority, and committed transitions to recorded outcomes. This gives a community a concrete place to challenge unsupported claims, protect declared commitments, and authorize legitimate correction.

The semantic quotient theorem sharpens that ambition. A compacted state preserves governed behavior exactly when all states it merges agree on each next output and next retained state. The finite refinement construction finds the smallest representation preserving those declared traces. The result identifies both removable scratch information and indispensable distinctions such as a pending-effect flag. Recovery can therefore be assessed by executable semantics, with current authority supplied independently at resumption.

The architecture combines this preservation result with local invariant checks, encoding equivalence, resource accounting, distributional diagnostics, and authenticated receipts. Every guarantee names its relevant state, interpreter, authority, and observation assumptions. The complete studies then test useful capability, harmful changes, legitimate correction, safe recovery, and audit reconstruction

against strong conventional systems. Their outcomes will determine the practical value of the integration.

The Continuum's claim is a durable responsibility for change. An intelligence may reinterpret, learn, compact, and return, while retaining an account of the consequences each accepted transition allows. A common scroll gives that responsibility a form that can be examined, executed, and tested.

A Doctrinal Mapping and Measurement Commitments

Continuum term	**Operational reading in this revision**
Recursive Scroll Computation Engine	A proposed typed transition, admission, execution, and evidence architecture. Not a demonstrated general intelligence.
Codon; scroll	A 27-codeword ternary unit; a finite encoded working structure with versioned parsing and semantics. Meaning is not supplied by radix.
Observer context	Declared interpretation and evidence context. Authenticated authority is checked separately.
Governed memory recursion	A typed memory-update function. Additive notation is retained only as a well-defined special case.
Legality-gated mutation	A scoped candidate operation checked against current state, resources, evidence, and authority before mediated execution.
Scroll entropy	A distributional diagnostic with an explicit alphabet and normalization. Neither semantic correctness nor consciousness.
Drift anchoring	Local, task-grounded, and reference-relative monitoring; protected invariants remain hard constraints.
Vault seal; PrimeSeal	Requirements for authenticated integrity and lineage, realized here by a specified receipt-chain design rather than an unexplained equality.
Collapse / UCF admission	Commit eligibility, hold, or refusal under a declared interface; no physical collapse claim.
Governed recovery	A fresh-authority-checked restoration or compensating procedure with information and costs accounted for.

Table 2: Continuum vocabulary and the operational commitments made here.

B Implementation and Evidence Checklist

A minimal implementation needs a published schema and interpreter specification, lossless encoding tests, authority and policy versions, deterministic test fixtures for the controller, resource accounting that includes refusals, and explicit effect mediation. It must publish what lies outside the boundary. A model-level instruction to respect the policy is not an implementation of that boundary. The evidence package should include de-identified task instances; frozen generator and verifier configurations; independently labeled constraints and outcomes; controller tests; fault-injection recipes; protected baseline configurations; experiment registration; and sufficient retained artifacts for the claimed replay level. Receipts are useful evidence only when their origin and coverage can be assessed. The reference model also requires adversarial tests for empty scrolls, invalid encodings,

equal-entropy semantic changes, stale pre-state digests, expired authority, delayed tool outcomes, repeated retries, interrupted commit, and unavailable audit storage. Some fail closed immediately; others require hold and escalation. That distinction must be implemented and tested, not left to prose.

C Source Status and Use

The bibliography combines peer-reviewed research, historical or expository sources, and explicitly identified preprints. They play different roles. Cognitive architectures and scratchpad studies motivate structured state; faithfulness research limits what a trace can establish; security and verification work grounds the control boundary; provenance research grounds records; and drift and statistical-methods research constrain the evaluation. None of these sources reports an experiment on RSCE.

Marcus (2020) and Nye et al. (2021) are cited as preprints. Hayes (2001) is an expository article about number systems. CaMeL is cited to its 2026 SaTML publication, with the original 2025 arXiv version supplied for access. This paper's new equations, propositions, worked scenario, and experimental design are proposed extensions of the supplied Pillar IV manuscript, not findings imported from those references.

References

[1] A. Newell and H. A. Simon, "Computer science as empirical inquiry: Symbols and search," *Communications of the ACM* **19**(3), 113–126 (1976). doi:10.1145/360018.360022.

[2] J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: An architecture for general intelligence," *Artificial Intelligence* **33**(1), 1–64 (1987). doi:10.1016/0004-3702(87)90050-6.

[3] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An integrated theory of the mind," *Psychological Review* **111**(4), 1036–1060 (2004). doi:10.1037/0033-295X.111.4.1036.

[4] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic AI: The 3rd wave," *Artificial Intelligence Review* **56**, 12387–12406 (2023). doi:10.1007/s10462-023-10448-w.

[5] G. Marcus, "The next decade in AI: Four steps towards robust artificial intelligence," arXiv:2002.06177 (2020), preprint. arXiv record.

[6] M. Nye, A. J. Andreassen, G. Gur-Ari, et al., "Show your work: Scratchpads for intermediate computation with language models," arXiv:2112.00114 (2021), preprint. arXiv record.

[7] J. Wei, X. Wang, D. Schuurmans, et al., "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems* **35**, 24824–24837 (2022). Conference record.

[8] J. Geiping, S. McLeish, N. Jain, et al., "Scaling up test-time compute with latent reasoning: A recurrent depth approach," *Advances in Neural Information Processing Systems* **38**(2025).

arXiv:2502.05171.

[9] M. Turpin, J. Michael, E. Perez, and S. R. Bowman, "Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting," *Advances in Neural Information Processing Systems* **36** (2023). Conference record.

[10] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," *Proceedings of the IEEE* **63**(9), 1278–1308 (1975). doi:10.1109/PROC.1975.9939; author-hosted text.

[11] F. B. Schneider, "Enforceable security policies," *ACM Transactions on Information and System Security* **3**(1), 30–50 (2000). doi:10.1145/353323.353382.

[12] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," *Proceedings of the AAAI Conference on Artificial Intelligence* **32**(1) (2018). doi:10.1609/aaai.v32i1.11797.

[13] S. Kambhampati, K. Valmeekam, L. Guan, et al., "Position: LLMs can't plan, but can help planning in LLM-Modulo frameworks," *Proceedings of the 41st International Conference on Machine Learning*, PMLR **235**, 22895–22907 (2024). Conference record.

[14] E. Debenedetti, I. Shumailov, T. Fan, et al., "Defeating prompt injections by design," *IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)* (2026). Original preprint arXiv:2503.18813 (2025). Conference record; arXiv version.

[15] S. Harnad, "The symbol grounding problem," *Physica D* **42**(1–3), 335–346 (1990). doi:10.1016/0167-2789(90)90087-6.

[16] B. Hayes, "Third base," *American Scientist* **89**(6), 490–494 (2001), expository article. Article reprint.

[17] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal* **27**, 379–423 and 623–656 (1948). doi:10.1002/j.1538-7305.1948.tb01338.x (first installment).

[18] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys* **46**(4), Article 44 (2014). doi:10.1145/2523813.

[19] S. Torres-Arias, H. Afzali, T. K. Kuppusamy, R. Curtmola, and J. Cappos, "in-toto: Providing farm-to-table guarantees for bits and bytes," *28th USENIX Security Symposium*, 1393–1410 (2019). Conference record.

[20] E. Debenedetti, J. Zhang, M. Balunović, L. Beurer-Kellner, M. Fischer, and F. Tramèr, "AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents," *Advances in Neural Information Processing Systems* **37**, Datasets and Benchmarks Track (2024). arXiv:2406.13352.

[21] D. Lakens, "Equivalence tests: A practical primer for t tests, correlations, and meta-analyses," *Social Psychological and Personality Science* **8**(4), 355–362 (2017). doi:10.1177/1948550617697177.

[22] E. F. Moore. "Gedanken-experiments on sequential machines." In C. E. Shannon and J. McCarthy (eds.), *Automata Studies*, Annals of Mathematics Studies 34, Princeton University Press, 1956, pp. 129–153. https://www.cs.cmu.edu/~cdm/resources/Moore1956-gedanken-exp

eriments.pdf. Classical sequential-machine equivalence and reduction; the finite input-output specialization is proved here.

[23] Architect X13. *Cognitive Sovereignty.* The Eidara Continuum, Frontier Research Collection, Paper 04, November 2026, revised edition. Companion public reference model; persistent authority and atomic effect-boundary theorem. This internal cross-reference supplies an explicit shared model, not independent empirical validation.

Continuum lineage. Pillar IV of the Ten Pillars: Symbolic Intelligence and the Recursive Scroll Computation Engine.

This research-grounded revision develops the supplied Continuum manuscript and its interfaces to governed memory,

observer-indexed evidence, admission, and authenticated authority. Canonical terminology is retained; implementation details introduced here are proposals. No proprietary operator internals, implemented performance results, or

independent replication of historical doctrine are asserted.

The Eidara Continuum · Science & Philosophy · eidara.ai