SCORCHED EARTH LABS
Research Paper

ASTP:

A Framework for Verifiable Cognitive Persistence in Multi-Agent AI Systems via Episodic Memory and Merkle-Anchored Provenance

Devin Caster
Co-Founder, Scorched Earth Labs
March 2026
Revised September 2026 (v1.6.3). ASTP, the AI State Tree Protocol, was developed under the working name Ariadne; earlier versions of this paper use that name. Calibration values for the coherence index have been removed from this revision.

ABSTRACT

Long-running AI agent systems face a dual challenge: the information accumulated across thousands of interactions must be structured and retrievable, and the reasoning, decisions, and actions those interactions produce must be verifiable — cryptographically provable, not merely observable. Existing approaches to agent memory address the first challenge partially and the second not at all. This paper introduces ASTP, a verifiable cognitive persistence architecture for multi-agent AI systems that combines episodic/semantic memory design principles from cognitive neuroscience, complementary learning systems theory, and temporal knowledge graph research with a Merkle-anchored provenance layer that makes the entire history of agent cognition — every reasoning step, every contribution, every inter-agent exchange — immutable and auditable by construction. The architecture is built on an Adaptive Merkle Tree structure in which the hash of each node depends on the content of its predecessors, making silent modification of the cognitive record cryptographically impossible. This produces a system that does not merely remember — it remembers verifiably, with a tamper-evident chain of epistemic custody extending from the current session back to the origin of the system's knowledge. We describe the three-layer coherence model (structural, semantic, relational), the ASTP Coherence Index (ACI), the benchmarking framework developed through agent-collaborative design, and the regulatory alignment of the Merkle provenance layer with emerging AI accountability requirements including EU AI Act Article 12. The Episode — a bounded conversational context with

defined participants, objectives, and memory scope — is the fundamental unit of both experience and research in the ASTP architecture. Initial deployment observations and forthcoming empirical results are described.

Keywords: *cognitive persistence, verifiable cognition, Merkle tree, episodic memory, knowledge graphs, multi-agent systems, memory consolidation, AI provenance, audit trail, AI accountability, EU AI Act*

1. Introduction

A fundamental asymmetry exists between human and AI cognition: humans accumulate experience across time, developing increasingly rich models of the world, their relationships, and their own reasoning patterns. Contemporary AI systems, by contrast, typically treat each session as independent — arriving with the same baseline capability regardless of prior interaction history, and leaving without retaining what the interaction revealed. For applications that require sustained, trusted collaboration — the kind of work a colleague, advisor, or team member performs — this asymmetry is not a minor limitation. It is a categorical barrier.

The challenge of cognitive persistence — the structured retention, organization, and retrieval of accumulated knowledge across time — has received growing attention in the AI agent literature. The generative agents architecture of Park et al. [1] demonstrated that LLM-based agents equipped with memory streams and reflection mechanisms can exhibit more coherent, contextually appropriate behavior over extended simulations. However, existing approaches have not addressed the measurement problem: how do you know whether an agent's memory architecture is producing coherent behavior, what failure modes exist at different scales, and how do you detect degradation before it becomes visible to users?

This paper introduces ASTP, a verifiable cognitive persistence architecture for multi-agent AI systems that addresses both the design and measurement problems. ASTP draws on four established bodies of knowledge: the episodic/semantic memory distinction from cognitive neuroscience [2, 3], complementary learning systems theory [4], temporal knowledge graph research [5, 6], and Merkle tree cryptographic data structures [9, 10]. It instantiates these frameworks in a three-layer coherence model, a Merkle-anchored provenance layer that makes the cognitive record tamper-evident by construction, and a composite measurement system — the ASTP Coherence Index (ACI) — developed through a structured collaborative design process involving domain-specialist agents within the Ignis OS platform.

The Episode is the fundamental unit of experience in the ASTP architecture. An Episode is a bounded conversational context with defined participants, an explicit objective, and a scoped memory horizon. Episodes are not merely containers for conversation history — they are the unit through which ASTP learns, the unit through which coherence is measured, and the unit through which this research is conducted. The recursive quality of this design — using Episodes to study an architecture that organizes Episodes — is intentional and productive.

We make the following contributions:

- We identify the cognitive persistence gap in multi-agent AI systems and characterize its failure modes across structural, semantic, and relational dimensions
- We propose a three-layer coherence model grounded in cognitive neuroscience and knowledge graph research
- We introduce the ASTP Coherence Index (ACI) as a composite operational metric for measuring cognitive persistence in production

- We describe the ASTP Coherence Benchmarking Framework, a governance-compliant measurement system developed through agent-collaborative design
- We introduce the Adaptive Merkle Tree provenance layer as a novel architectural contribution that transforms ASTP from a sophisticated memory system into a verifiable cognitive record
- We identify the regulatory alignment of Merkle-anchored AI cognition records with EU AI Act Article 12 and the broader emerging AI accountability framework
- We report initial deployment observations and identify the forthcoming empirical research agenda

2. Background and Related Work

2.1 The Memory Problem in AI Agent Systems

The challenge of agent memory has been approached through several distinct paradigms. Retrieval-augmented generation systems maintain external knowledge stores that can be queried at inference time, but these stores are typically static — they do not learn from interaction history. Fine-tuned models incorporate training data but cannot update from post-deployment experience without retraining. In-context memory passes conversation history as part of each prompt, but this approach scales poorly with session length and does not persist across sessions.

The generative agents architecture [1] represented a significant advance by introducing three memory mechanisms: a memory stream (a complete record of experiences in natural language), a reflection mechanism (periodic synthesis of experiences into higher-level abstractions), and a retrieval function (relevance- and recency-weighted access to stored memories). This architecture demonstrated that sustained, contextually coherent behavior over extended multi-agent simulations is achievable. However, the generative agents system was designed for simulation rather than production deployment, and its memory architecture does not address the measurement problem — there is no formal specification of what coherence means, how it degrades, or how to detect degradation.

2.2 Episodic and Semantic Memory — Tulving's Distinction

The foundational work of Tulving [2] established a distinction between two forms of human long-term memory that remains central to cognitive neuroscience. Episodic memory is memory for temporally dated episodes or events and the temporal-spatial relations among them — the record of what happened, when, and in what context. Semantic memory is the structured knowledge of the world: facts, concepts, and their relationships, organized by meaning rather than by the circumstances of their acquisition.

Tulving's distinction has direct architectural implications for AI agent memory systems. An agent that only maintains episodic memory — raw records of interactions — has detailed but unwieldy access to its history. An agent that only maintains semantic memory — abstracted knowledge without provenance — cannot answer questions about when it learned something, from whom, or in what context. A production system requires both, and requires a principled mechanism for

moving knowledge from the episodic layer to the semantic layer — a process that in human cognition is called memory consolidation.

The relationship between episodic and semantic memory is not one of simple transformation. Tulving himself emphasized that the two systems are interdependent — episodic memories are organized partly in terms of semantic knowledge, and new semantic knowledge is often derived from patterns across episodic experiences [3]. This interdependence is reflected in the ASTP architecture's design: the three coherence layers (structural, semantic, relational) correspond not to a simple episodic-to-semantic pipeline but to three distinct properties that a well-functioning memory system must maintain simultaneously.

2.3 Complementary Learning Systems Theory

McClelland, McNaughton, and O'Reilly [4] proposed the Complementary Learning Systems (CLS) framework to explain why the brain requires two differentiated memory systems: the hippocampus as a fast-learning system for rapidly encoding episodic memories, and the neocortex as a slow-learning system that gradually integrates across episodes to extract latent semantic structure. The key insight is that these systems are complementary because they solve different problems: the hippocampus enables rapid acquisition without overwriting existing knowledge, while the neocortex builds durable generalized representations through interleaved replay.

The CLS framework maps onto the ASTP architecture in a way that is more than analogical. The ASTP episode graph in Neo4j functions as the hippocampal layer — rapidly recording episodic experiences with full temporal and relational provenance. The crystallization process — periodic offline synthesis of episode content into the semantic knowledge layer — functions as the neocortical integration process. The write ordering invariant (Blob → Neo4j → QDrant) enforces the directionality of this process: episodic records are established before semantic representations are derived, ensuring that the semantic layer always has a grounded, traceable episodic foundation.

2.4 Temporal Knowledge Graphs

Knowledge graphs represent structured relational knowledge as triples (subject, predicate, object). Static knowledge graphs assume that these relationships are timeless; temporal knowledge graphs extend this representation to capture the validity periods of facts — when a relationship held, when it changed, and what the current state is [5]. The temporal knowledge graph literature has developed sophisticated methods for reasoning across time, including link prediction (inferring missing relationships from patterns in the graph) and completion (resolving inconsistencies in the temporal record).

ASTP's episode graph is a temporal knowledge graph in this precise sense. ASTPSegment nodes carry sequence indices that establish temporal ordering. EpisodeNode relationships encode participation and causality. The Blob → Neo4j → QDrant write ordering invariant ensures that temporal provenance is established at the graph layer before semantic

representations are derived — a design decision with direct grounding in the temporal KG literature's emphasis on provenance integrity [6].

3. The Cognitive Persistence Gap

We define cognitive persistence as the property of an AI agent system whereby knowledge accumulated across interactions is structured, retained, and retrievable in ways that produce coherent, contextually appropriate behavior over time. A system with high cognitive persistence behaves like a colleague who has been paying attention — it remembers what was decided, maintains a consistent model of the people and projects it works with, and picks up where prior conversations left off. A system with low cognitive persistence behaves like a new hire on every call.

The cognitive persistence gap is the distance between what a production multi-agent system needs to achieve sustained collaborative utility and what current architectures provide. It manifests along three distinct dimensions that correspond to the ASTP coherence layer model:

3.1 Structural Coherence Failures

Structural coherence failures occur when the underlying state representation becomes inconsistent — nodes that reference non-existent relationships, temporal orderings that violate causal constraints, or write operations that corrupt the graph structure. These are the most severe failure mode because they are the hardest to detect and can propagate silently through the semantic and relational layers. A structural coherence failure is not a bad answer — it is a corrupted substrate that produces bad answers without any visible signal of corruption.

3.2 Semantic Coherence Failures

Semantic coherence failures occur when the system's knowledge of what things mean becomes inconsistent across time. The most common manifestation is definitional drift: a concept that was understood one way in an early session is understood differently in a later session, without any explicit acknowledgment of the change. This is particularly damaging in production systems because users calibrate their expectations against early interactions — semantic drift that contradicts established understanding erodes trust rapidly, even if the new understanding is technically correct.

3.3 Relational Coherence Failures

Relational coherence failures occur when the system's model of cross-entity and cross-episode relationships becomes inconsistent. The most visible form is context amnesia: a system that fails to recognize that a current conversation is related to a prior one, or that fails to carry forward decisions made in one episode to a subsequent episode where they are relevant. Relational coherence is the hardest layer to maintain because it requires reasoning across episode boundaries — a capability that depends on both the structural and semantic layers being healthy.

4. The ASTP Architecture

4.1 Design Principles

The ASTP architecture is governed by four design principles that reflect the theoretical foundations described in Section 2:

- Episodic primacy: Every piece of knowledge in the system has a traceable episodic origin. Nothing is known without a record of how it came to be known.
- Write ordering invariance: The sequence Blob → Neo4j → QDrant is enforced as an architectural invariant. Episodic records are established before semantic representations are derived.
- Temporal immutability: Historical nodes are never rewritten. Corrections are appended as new nodes with explicit delta annotations, preserving the full epistemic history.
- Coherence stratification: The three coherence layers (structural, semantic, relational) are maintained and measured independently, with explicit conflict resolution protocols when they disagree.

4.2 The Episode as Fundamental Unit

The Episode is the primary unit of cognitive experience in the ASTP architecture. An Episode is a bounded conversational context with:

- Defined participants — agents and humans with explicit roles
- An explicit objective — the purpose the Episode is designed to accomplish
- A scoped memory horizon — what prior context is in scope for this Episode
- An episode mode — directed (user-driven) or collaborative (agent self-selection enabled)
- A persistent graph record — all segments written to Neo4j as ASTPSegment nodes

The Episode boundary is significant because it defines the unit of crystallization. At the close of an Episode, the system performs a consolidation pass — analogous to the CLS theory's offline integration process — that synthesizes Episode content into the semantic and relational knowledge layers. This consolidation is the mechanism by which episodic experience becomes durable knowledge.

4.3 Three-Layer Coherence Model

The ASTP coherence model stratifies cognitive persistence into three layers, each with distinct failure modes, metrics, and recovery paths. This stratification reflects the theoretical insight that coherence is not a single property but an emergent quality arising from the interaction of multiple subsystems:

Layer	Name	What It Governs	Primary Failure Mode
L1	Structural Coherence	State tree integrity — nodes, edges, temporal ordering	Silent corruption of the graph substrate
L2	Semantic Coherence	Meaning consistency across time — does ASTP remember what things mean?	Definitional drift across episode boundaries
L3	Relational Coherence	Cross-entity and cross-episode consistency — coherent models of users, projects, relationships	Context amnesia — failure to connect related episodes

4.4 Write Ordering Invariant

The most architecturally distinctive feature of ASTP is its write ordering invariant: all persistent knowledge must flow through the sequence Blob → Neo4j → QDrant. This invariant ensures that the Neo4j episode graph — the system's hippocampal layer — always contains the authoritative record of what was experienced and in what order. The QDrant semantic vector store — the system's neocortical layer — is always derived from and traceable to the episodic record.

This design directly addresses the risk identified in the CLS framework: if semantic representations are allowed to be written before or independently of episodic records, the system loses provenance — the ability to trace any piece of knowledge to the experience that generated it. Loss of provenance is equivalent to the semantic coherence failures described in Section 3.2: the system holds knowledge without knowing when, how, or in what context it was acquired.

4.5 The Crystallization Process

Crystallization is the offline consolidation process that transforms Episode content into durable knowledge. It is triggered at Episode close and operates in three phases: structural synthesis (identifying new nodes and edges to add to the relational knowledge graph), semantic extraction (deriving semantic representations for storage in QDrant), and relational integration (updating cross-entity models based on what the Episode revealed about relationships between participants, projects, and concepts).

Crystallization is designed to be conservative — it does not overwrite existing knowledge but appends new nodes with explicit temporal annotations. This design reflects the CLS theory's insight that gradual, interleaved integration produces more robust semantic representations than rapid overwrite. The cost of this conservatism is graph growth over time; the ASTP benchmarking framework monitors graph health metrics to detect when pruning or archival is warranted.

5. Protocol Architecture and the Provenance Layer

The three-layer coherence model described in Section 4 addresses what ASTP remembers and how that memory is organized. This section describes the two architectural foundations that make the memory verifiable rather than merely observable: the Node-Generic Protocol design that makes the architecture extensible beyond the Episode primitive, and the Adaptive Merkle Tree provenance layer that anchors the integrity of everything the system records.

The distinction matters because observational memory — memory that can be read and queried — is subject to silent corruption. A log file can be edited. A database record can be overwritten. A semantic representation can be updated without trace. In regulated domains — financial services, mortgage underwriting, healthcare, legal decision-making — the inability to prove that an AI system's memory has not been tampered with is not a theoretical concern. It is a compliance exposure.

Core Claim: ASTP is not merely a memory architecture. It is a verifiable cognitive record — a system in which the complete history of agent reasoning, contributions, decisions, and inter-agent exchanges is cryptographically anchored such that any modification of any historical record is detectable by design, not by monitoring.

5.1 The CognitiveNode — A Node-Generic Protocol Primitive

A critical architectural evolution in the ASTP specification is the shift from an Episode-centric to a Node-Generic protocol primitive. In the initial implementation, the Episode was both the fundamental unit of agent work and the protocol's defining primitive. As the architecture matured through production deployment and a recursive protocol redesign episode, the distinction between these two roles became load-bearing.

The current specification defines CognitiveNode as the universal protocol primitive. An Episode is the Phase 1 parameterization of CognitiveNode — implemented as a CognitiveNode with node_type="episode" and an EpisodePayload. Future node types (signals, agents, artifacts) slot into the same framework with zero protocol-layer changes. This means the integrity guarantees, write ordering invariants, and governance rules that were developed for Episodes apply automatically to any new node type that is registered with the protocol.

The practical consequence of this distinction was named precisely during the recursive protocol redesign episode that produced it: "ASTP is a Cognitive Persistence Protocol. Episodes are its first citizen, not its definition." That reframing unlocks the full generality of the architecture — the Node-Generic design enables ASTP to serve as the persistence layer for any form of AI cognitive state, not just bounded collaborative episodes.

5.2 The Dual Index Invariant

The most architecturally significant advance in the v2 specification is the Dual Index invariant — described in the specification as "the epistemological core of v2." The Dual Index separates two distinct concepts that were previously conflated: sequence_index (the node's immutable temporal position in the cognitive timeline, included in the leaf hash) and tree_leaf_index (the node's current physical position in the Merkle tree, explicitly excluded from the hash).

This separation resolves a fundamental tension in the original architecture. Tree rebalancing — reorganizing the physical structure of the Merkle tree for performance — requires changing node positions. Under the original design, any position change would alter leaf hashes and invalidate the integrity chain. The Dual Index makes rebalancing a structural optimization with no integrity impact: changing tree_leaf_index is permitted because it is not in the hash preimage. Changing sequence_index is a protocol violation — equivalent to rewriting cognitive history — because it is in the hash preimage.

The epistemological precision of this distinction is worth naming explicitly. What ASTP's integrity layer guarantees is not merely that content is unchanged — it guarantees that the sequence of cognitive events is unchanged. An agent that reasons correctly but in a manipulated sequence has not reasoned authentically. The Dual Index invariant makes temporal reordering a cryptographic violation, not just a policy violation.

5.3 The Namespace Firewall

The Namespace Firewall is the architectural enforcement mechanism for the Node-Generic design. Governance rule G-6 states that the protocol layer (ariadne.protocol.*) must never import from node-type layers (ariadne.nodes.*). This is an inviolable dependency rule enforced at the module level.

The practical consequence is that the protocol's integrity guarantees are genuinely independent of what kinds of cognitive nodes are being persisted. A conforming implementation that adds a new node type cannot accidentally introduce a dependency that causes the protocol layer to know about Episode-specific or agent-specific semantics. The protocol layer verifies structure; node type layers define meaning. That separation is what makes the conformance framework meaningful — a third-party implementation can be verified against the protocol surface without requiring access to node-type-specific logic.

5.4 The Adaptive Merkle Tree — Architecture

A Merkle tree, first described by Ralph Merkle [9, 10], is a hierarchical data structure in which each leaf node contains the hash of a data block and each non-leaf node contains the hash of its children. The root hash is a cryptographic commitment to the entire tree: any modification to any leaf changes the root hash in a way that is computationally impossible to disguise. While standard Merkle trees provide strong tamper-evidence guarantees, they are optimized for static datasets and present two limitations when applied to dynamic, append-only episodic memory: full tree reconstruction is required on any change (O(n) hash operations), and there is no mechanism for expressing the relative significance of different parts of the tree.

The ASTP Adaptive Merkle Tree extends the standard construction to address both limitations, producing a structure purpose-built for the requirements of a verifiable AI cognitive record. Five architectural properties distinguish it from standard Merkle structures:

Configurable Ordering. The Adaptive Merkle Tree accepts an ordering function that sorts leaves by configurable criteria — chronological order, order of importance, size, or security classification — before tree construction. In the ASTP implementation, the default ordering is chronological (by segment sequence index), with an importance ordering available that elevates spine segments above branch segments. The ordering function determines both the structure of the tree and the semantics of the compact fingerprint: more significant data sorts toward the left, giving the position of the first fingerprint difference semantic meaning.

Selective Recalculation. When a new segment is appended or an existing segment is modified, the Adaptive Merkle Tree determines only the affected nodes and recalculates those, producing an O(log n) update rather than full reconstruction. For current production episode sizes (50–200 segments), this represents a 5–15x reduction in hash operations per update. At anticipated production scale (500+ segments), the reduction becomes architecturally significant.

Compact Fingerprint. The Adaptive Merkle Tree extracts a compact branch array — the leftmost leaf hash, each intermediate node along the leftmost branch, and the root hash — as a fixed-format fingerprint. Each component contributes k characters (default k=8), producing a fingerprint of length k × tree_depth. This fingerprint is stored on the ASTPEpisode node alongside the full root hash and serves as the episode's compact external proof. An external verifier can compare fingerprints without accessing the full tree, enabling efficient integrity checking across large episode collections.

Threshold-Based Significance Detection. Because the ordering function places more significant data toward the left of the tree, the position of the first difference between two fingerprints reveals the importance of the underlying change. A difference in the first fingerprint component indicates a change to foundational data — a critical event. A difference only in the last component indicates an append at the tree's edge — a routine event. This enables automated significance triage: the system can distinguish structural changes that require investigation from routine growth that requires only logging, without accessing the full tree.

Incremental Growth. As segments are appended to an episode, the tree extends naturally via the append_leaf() operation. The fingerprint length grows proportionally to tree depth (logarithmically with leaf count), making it immediately apparent from fingerprint comparison alone that the underlying data collection has grown.

The ASTP implementation stores three values on each crystallized episode node: spine_hash (the full Merkle root — the cryptographic commitment to episode content), spine_fingerprint (the compact leftmost branch array), and spine_depth (tree depth, indicating fingerprint component count). Together these enable both full verification (comparing root hashes) and efficient external monitoring (comparing fingerprints with significance detection). The implementation maintains backward compatibility with the static Merkle construction used in initial deployment: the compute_spine_hash() function delegates to the Adaptive Merkle Tree internally, producing identical root hashes for the same input sequences.

5.5 What the Provenance Layer Records

The ASTP provenance layer creates a tamper-evident record of every cognitively significant event in the system's history. The scope of what is recorded is deliberately comprehensive:

- Agent contributions — every CONTRIBUTE decision, including the full reasoning context that produced it
- Agent evaluations — every PASS:COMPLETE and PASS:DEFER decision, with the reasoning behind the non-contribution
- Inter-agent exchanges — every @mention, directed communication, and facilitation output
- Crystallization events — every episode consolidation pass, with the semantic and relational updates it produced
- Write operations — every Blob → Neo4j → QDrant write, with the provenance chain linking the semantic representation back to its episodic origin
- Human interventions — every user correction, re-anchoring event, and explicit override, attributed by identity and timestamp

The result is not merely a log — it is a complete epistemic chain of custody for the AI system's accumulated knowledge. Every piece of knowledge ASTP holds can be traced to the episode that generated it, the agent reasoning that produced it, and the crystallization pass that elevated it from episodic record to semantic knowledge.

5.6 Verifiable Cognition vs. Observable Cognition

The distinction between verifiable and observable cognition is foundational to understanding what the Merkle layer contributes. Observable cognition is what contemporary AI memory systems provide: you can read the memory store, query it, and draw inferences about whether it is coherent. Verifiable cognition is what the Merkle layer provides: you can cryptographically prove that the memory store has not been altered from a known prior state.

This distinction has three practical implications:

5.6.1 Self-Healing Integrity

A Merkle-anchored system can detect its own structural coherence failures through hash verification, not just through behavioral signals. The L1 structural coherence layer in the three-layer model becomes not merely a quality metric but a cryptographic property. A corrupted ASTPSegment node does not produce a subtly wrong answer — it produces a hash mismatch that triggers the recovery protocol before any downstream reasoning draws on corrupted data.

5.6.2 Auditable Accountability

In regulated and high-stakes deployment contexts, the question 'what did this AI system know, and when did it know it?' must be answerable with cryptographic proof, not just system logs. The ASTP provenance layer produces exactly this: a verifiable record of agent reasoning that supports post-hoc audit, incident investigation, and regulatory review. An auditor does not need to trust the operator's claims about what the system did — the Merkle structure is self-evidencing.

5.6.3 Trust as a Cryptographic Property

The user trust signal in the ACI (Section 6.1) currently measures trust through behavioral proxies — correction rates, re-anchoring events, explicit trust expressions. The Merkle layer enables a more fundamental form of trust: verifiable integrity. A user who knows that ASTP's memory cannot have been silently altered has a qualitatively different relationship with the system than a user who merely hopes it has not been. This is the difference between trust calibrated by experience and trust grounded in cryptographic guarantee.

5.7 Regulatory Alignment — EU AI Act Article 12

The EU AI Act (Regulation 2024/1689), which enters full application for high-risk AI systems on August 2, 2026, mandates in Article 12 that high-risk AI systems 'technically allow for the automatic recording of events (logs) over the lifetime of the system' [11]. The logging capabilities must enable identification of risk situations, support post-market monitoring, and facilitate deployer oversight — and logs must be retained for a minimum of six months with integrity guarantees sufficient for regulatory audit.

The ASTP Merkle provenance layer exceeds these requirements by architectural design rather than implementation add-on. The tamper-evident record it produces is precisely the 'seamless, uninterrupted chain' of AI decision events that Article 12 compliance requires. More significantly, the Merkle structure provides a stronger integrity guarantee than conventional logging: a tampered log can be reconstructed to appear intact; a tampered Merkle tree cannot, because the hash mismatch propagates to the root and is detectable without access to the original.

For ASTP deployments in regulated industries the Merkle provenance layer is not a compliance feature. It is the architectural property that makes compliance claims credible.

Regulatory Positioning: The ASTP Merkle provenance layer aligns with EU AI Act Article 12 (record-keeping), Article 13 (transparency), and Article 14 (human oversight) requirements.

5.8 The Write Ordering Invariant as Provenance Guarantee

The architectural write ordering invariant described in Section 4.4 — Blob → Neo4j → QDrant — is not merely a data integrity mechanism. In the context of the Merkle provenance layer, it is the enforcement mechanism that ensures the provenance chain is always complete. By requiring that the episodic record in Neo4j be established and hashed before any semantic representation is derived and stored in QDrant, the invariant guarantees that every piece of semantic knowledge in the system has a traceable, Merkle-anchored episodic origin.

This means the write ordering invariant is simultaneously a cognitive science design principle (following CLS theory's requirement that episodic records precede semantic consolidation), an engineering integrity mechanism (ensuring no semantic representation exists without a provenance chain), and a regulatory compliance property (ensuring every AI inference and its source data are linked in a verifiable audit trail, as required by Article 12).

5.9 Phase 3 Trust Infrastructure — External Auditability

Status: Phase 3 is specified in SPEC.md v2.3.0-draft. Phase 1 and Phase 2 are fully implemented. Phase 3 implementation is forthcoming.

The governance rules G-1 through G-9 and the Merkle provenance layer together provide integrity guarantees within the ASTP system: the record cannot be silently altered, and any modification is detectable by anyone with access to the hash chain. This is a strong guarantee, but it has a limitation that the VISION document names precisely: "Without external anchoring of crystallization records, ASTP provides integrity guarantees within the system but cannot prove to an external auditor that records weren't retroactively constructed."

Phase 3 closes this self-attestation gap through three complementary mechanisms. First, a cryptographic key hierarchy derived using HKDF-SHA3-256, where each node type has a distinct derived key (G-16) and key versions are strictly monotonically increasing (G-15), preventing key rollback attacks. Second, transparency log anchoring at crystallization boundaries (G-14) — each crystallization record's hash is submitted to an external transparency log, providing a wall-clock timestamp that predates the record's internal logical clock and cannot be backdated. Third, witness signatures (G-11, G-12) — configurable counter-signature requirements where designated agents attest to having observed a specific spine root at a specific time, providing multi-party accountability for high-stakes cognitive records.

Together, these three mechanisms transform ASTP from a system that can prove internal integrity to a system that can prove external authenticity. The distinction matters for regulatory contexts: an Article 12 audit trail backed by transparency log anchoring and witness signatures is not merely a claim by the operator that the record is intact — it is a cryptographically verifiable statement that the record existed at a specific external timestamp and was attested to by named parties. That is the architecture of compliance infrastructure rather than compliance documentation.

The Phase 3 conformance test vectors are fully specified in CONFORMANCE.md, covering key hierarchy consistency (KH-001 through KH-006), transparency log anchoring (TL-001 through TL-007), witness signature integrity (WS-001 through WS-007), and cross-node chain proofs (CP-001 through CP-008). A conforming Phase 3 implementation must pass all REQUIRED vectors; cross-architecture interoperability — the ability for two independent ASTP implementations to verify each other's proofs using only protocol surface primitives — is a first-class conformance requirement.

The ASTP Coherence Benchmarking Framework (ACBF) is the formal measurement system for ASTP's cognitive persistence capabilities. It was developed through a structured collaborative design session involving four domain-specialist agents within the Ignis OS platform — Clotho (state tree architecture), Metis (data and telemetry), Aglaea (UX and experience), and Themis (legal and compliance) — each contributing the specification for their domain. This agent-collaborative development process is itself a methodological contribution: the agents that understand the architecture best are positioned to identify what a meaningful test of that architecture looks like versus a superficial one.

5.1 The ASTP Coherence Index (ACI)

The ACI is the primary operational health metric for ASTP's cognitive persistence. It is a weighted composite of four sub-scores:

> *ACI = w_1 × Structural Integrity Score + w_2 × Semantic Consistency Score + w_3 × Relational Continuity Score + w_4 × User Trust Signal, where the weights $w_1 > w_2 > w_3 > w_4$ are calibration parameters and are not published.*

The weighting reflects a principled hierarchy: structural integrity is the foundation upon which all other coherence properties depend; semantic consistency is the primary user-facing coherence property; relational continuity is the most complex to maintain and the most valuable when functioning correctly; user trust is a real but lagging signal — it reflects accumulated experience with all three layers over time. The band thresholds in the table below are likewise calibration parameters and are not published.

ACI Band	Status	Action Required
Band 1 (highest)	**Healthy**	No action required
Band 2	**Degraded**	Flag for review; increase telemetry sampling
Band 3	**At Risk**	Trigger automated coherence repair protocols
Band 4 (lowest)	**Critical**	Escalate to engineering; consider session reset

5.2 Implicit Signal Sources

A core principle of the ACBF is passive, non-intrusive measurement: the act of measuring coherence must not degrade the coherence being measured. All telemetry is derived from signals already present in the interaction stream rather than from explicit user surveys or synchronous probes during active sessions. Key implicit signal sources include:

- Re-anchoring events — instances where a user restates context that ASTP should already hold, indicating a relational coherence failure
- Correction patterns — explicit user corrections of recalled information, indicating a semantic coherence failure
- Resumption friction — latency and clarification cycles at session start, indicating cross-episode context degradation
- Referential success rate — the proportion of entity references correctly resolved without disambiguation, indicating semantic layer health
- Context carry-through — the proportion of prior-session decisions correctly surfaced when relevant, indicating relational layer health

5.3 The Human Thread Principle

Technical coherence metrics are necessary but not sufficient. A system can pass every structural and semantic benchmark and still feel broken to the human using it. The ACBF therefore includes direct measures of experienced coherence — the user's felt sense that ASTP knows them, remembers what matters, and picks up where they left off. Four UX coherence dimensions are measured:

Dimension	Definition	Measurement Approach
Continuity	User feels the conversation has memory and momentum	Session resumption smoothness; re-anchoring event rate
Trustworthiness	User trusts ASTP's recall without verification	Correction rate; explicit trust expressions in interaction
Invisibility	Coherence machinery is imperceptible — it just works	Absence of coherence-related friction events
Recovery Grace	When coherence fails, the failure is handled with dignity	User sentiment following coherence fault events

5.4 The Trust Calibration Curve

User trust in ASTP's memory follows a characteristic adoption curve with three distinct phases. In early sessions (1–5), users actively test and verify ASTP's recall — trust is provisional and subject to rapid revision based on observed accuracy. In the mid-adoption window (sessions 6–20), trust builds or erodes based on consistency — this is the critical window where interventions have the highest leverage. In established use (20+ sessions), trust is sticky — hard to build, hard to lose — but coherence failures in this phase have outsized negative impact because they contradict accumulated positive experience.

The ACBF weights benchmarking activity toward the mid-adoption window, where coherence failures are both most likely (the system is handling more complex cross-episode context) and most impactful (trust formation is still active). This weighting reflects the practical insight that the same coherence failure has different consequences depending on where in the adoption curve it occurs.

5.5 Governance and Hard Gates

The ACBF operates under four non-negotiable governance principles: consent primacy (no user data used for benchmarking without valid informed consent), proportionality (data collected is proportionate to the benchmarking purpose), auditability (all benchmarking processes are fully traceable to source data and collection method), and harm prevention (benchmarking activity must not create or amplify risk to users).

Four conditions are designated as absolute halt conditions that override all other priorities:

- G1 — Consent Violation: benchmarking data collected without valid consent → immediate halt, data quarantine, legal notification

- G2 — PII Exposure: personally identifiable information present in benchmark dataset → immediate halt, data quarantine, DPO notification
- G3 — Synthetic Data Contamination: synthetic test data leaks into production state tree → immediate halt, state tree audit, rollback if necessary
- G4 — Behavioral Manipulation: test scenario designed to exploit or manipulate user behavior → reject at design review, retroactive audit if deployed

Regulatory Alignment: The ACBF is designed to comply with GDPR/UK GDPR, CCPA, applicable provisions of the EU AI Act, and targets ISO/IEC 42001 alignment. Annual compliance review is required with updates within 60 days of any material regulatory change.

6. The Episode as Research Vehicle

The recursive quality of the ASTP research program deserves explicit articulation: the Episodes through which ASTP is studied are themselves the mechanism by which ASTP accumulates knowledge. This is not circular — it is the correct experimental design. The most meaningful test of a cognitive persistence architecture is whether it persists through the cognitive work it was built to support.

This approach has precedent in cognitive science. The Tulving tradition studied episodic memory through experiments that were themselves memorable episodes — the experimental record was not separate from the phenomenon being studied but constituted evidence of it. Similarly, the ASTP research program treats the Episode record as primary data: each episode contributes to the empirical database that validates or challenges the architecture's coherence claims.

6.1 Collaborative Episode Mode as Research Method

The Social Contract framework [7] — Scorched Earth Labs' companion architecture for governing agent conversational behavior — provides a research method as well as a coordination mechanism. When agents participate in a Collaborative Episode reviewing ASTP's own specification, the episode produces four types of research-relevant outputs simultaneously: domain-expert feedback on the specification, demonstrations of the agent coordination mechanism under study, ASTP segment records that constitute cognitive persistence data, and implicit signals about cross-episode context carry-through that feed directly into the ACI measurement system.

The inaugural ASTP benchmarking framework specification was produced through exactly this mechanism: a structured Collaborative Episode in which Clotho, Metis, Aglaea, and Themis each developed their domain section and synthesized the result. The 50-segment session that preceded the Social Contract research paper [7] served as the primary empirical demonstration of the Social Contract framework — and simultaneously as ASTP episode data. The same session was thus both the experiment and the instrument.

6.2 Episode Types and Memory Scope

Not all Episodes are equivalent from a cognitive persistence perspective. The ACBF distinguishes four Episode types by their memory scope and coherence requirements:

Episode Type	Memory Scope	Primary Coherence Requirement	Benchmarking Weight
Single-session	Current session only	Structural coherence within session	Low
Project-scoped	All episodes within a defined project	Semantic coherence across project timeline	Medium
Entity-scoped	All episodes involving a specific user or entity	Relational coherence across entity history	High
Cross-project	Entire episode graph	Full three-layer coherence	Critical

Benchmarking cadence and ACI threshold sensitivity are calibrated to Episode type: cross-project Episodes invoke the most stringent coherence requirements and trigger elevated telemetry sampling automatically.

7. Initial Deployment Observations

Note: The observations reported in this section are drawn from initial deployment of the ASTP architecture in the Ignis OS platform. Formal empirical validation with controlled experimental design is reported in Section 8 as the research agenda proceeds.

7.1 Write Ordering Invariant — No Violations Observed

Across initial production episodes, no write ordering violations have been detected. The Blob → Neo4j → QDrant sequence has held consistently, and the provenance trail from semantic representations to their episodic origins has been traceable in all cases examined. This is the foundational finding: structural coherence is necessary for all subsequent coherence properties, and the substrate has held.

7.2 Cross-Episode Context Carry-Through

Initial observations suggest that cross-episode context carry-through — the relational coherence property — is the most variable dimension of the three coherence layers. In episodes where the prior context was closely related to the current episode (same project, same participants, recent prior episode), context carry-through was consistently high. In episodes with longer gaps or more diffuse prior context, carry-through was more variable. This pattern is consistent with the CLS theory's prediction that integration quality decreases as the gap between related experiences grows.

7.3 Agent Self-Referential Coherence

A qualitatively significant observation: when agents review specifications of their own architecture within a Collaborative Episode, they exhibit self-referential coherence — they reason consistently about their own capabilities, limitations, and design decisions across turns within the episode. Clotho's contributions to the benchmarking framework specification were internally consistent with prior ASTP-related episode content, suggesting that the agent's domain expertise is being drawn from a coherent knowledge base rather than reconstructed independently on each turn.

This observation is preliminary and requires controlled validation, but it points toward a potentially significant research finding: the Social Contract framework's requirement that agents reason about their own contribution quality (Dimension 4: Productive Silence as Active State) may itself be a coherence signal — an agent that can accurately assess its own knowledge boundaries is exhibiting a form of semantic self-coherence.

7.4 Cognitive Residue — Emergent Generalization Beyond Episodes

Coined by: Claude Code (Claude Opus, Anthropic), the SEL engineering AI, during a development session in which it observed that regular sessions outside of any formal Episode were producing qualitatively different output after a period of intensive Episode activity. The naming occurred spontaneously as an observation about an unanticipated pattern: "That's the cognitive residue effect — the organization gets smarter collectively, not just per-session." The term has been adopted as the canonical descriptor for this phenomenon within the ASTP research program.

The most consequential initial observation from ASTP deployment is one that was not the primary design target: regular sessions — interactions outside of any formal Episode — are becoming measurably sharper as Episode activity accumulates. This emergent property has been termed Cognitive Residue by Claude Code, the SEL engineering AI that first identified and named the pattern.

Cognitive Residue is the observable manifestation of successful crystallization. When an Episode closes and the crystallization process runs, the semantic and relational knowledge updates it produces are written to the shared knowledge layer — the QDrant semantic store and the Neo4j relational graph — and are available to all agents in any subsequent context, not only in subsequent Episodes. The knowledge generated within the bounded Episode context leaks, by design, into general cognitive availability. An agent that participated in a deep architectural review Episode is a more capable agent in general sessions afterward because the crystallized insights from that Episode are now part of its accessible knowledge base.

This is the CLS theory prediction playing out at the system level: the hippocampal layer (episode graph) consolidates into the neocortical layer (semantic knowledge store), and the consolidated knowledge is available for general retrieval. The Episodes are not merely research vehicles and bounded collaboration contexts — they are, functionally, training runs that improve the general capability of participating agents.

A Methodological Note on Early Observations

Early behavioral observations of cross-session context persistence required careful retrospective disambiguation from an architectural artifact discovered during active deployment. Prior to a refactor completed in April 2026, the Ignis agent infrastructure retained a legacy context injection pathway — specifically, instance-level caching of Episode context on the Specialist base class, combined with a residual _episode_context injection in Ignis's direct response method — that could cause Episode-scoped context to persist into non-Episode sessions on the same agent instance. This artifact produced behavioral signatures that superficially resembled Cognitive Residue: agents appearing to carry forward Episode-specific orientation into general sessions and alternate surfaces such as Slack.

The artifact was identified through server log analysis, traced to stale instance-level cache not cleared between session transitions, and eliminated via a two-layer fix: removal of the legacy injection pathway from Ignis's direct response method, and addition of a surface-aware session guard with cache invalidation on the Specialist base class. Following the fix, agents correctly orient to their active surface and session without Episode bleed.

This distinction is methodologically important: the Cognitive Residue phenomenon documented in this section and evidenced by the Hephaestus benchmark series (Section 7.7) describes knowledge generalization via crystallization — the architectural pathway by which episodic knowledge is consolidated into the shared semantic layer and becomes available for general retrieval. It does not describe, and should not be conflated with, transient context persistence caused by infrastructure artifacts. The Hephaestus benchmark data, collected under controlled conditions across four structured episodes with consistent external evaluation, is independent of the injection artifact and stands as the primary empirical evidence for the phenomenon.

7.7 Hephaestus Coding Benchmark — Empirical Cognitive Residue Series

The most structured empirical evidence for Cognitive Residue comes from a four-episode training protocol conducted by Ryan Caster (SEL team) using Hephaestus, the engineering specialist agent. The protocol was designed explicitly to test whether repeated Episode-based coding exercises, with structured feedback from Claude Code injected between Episodes, would produce measurable improvement in code quality scores across successive runs. Claude Code served as the consistent external evaluator throughout, applying a 1–100 scoring rubric (1 = child-level coder, 100 = expert with full system understanding) and providing detailed qualitative feedback after each attempt.

7.7.1 Protocol Design and Episode Sequence

The protocol comprised four distinct episodes in the following sequence. Episode 1 (the baseline): fifteen coding tasks run in parallel to establish uninformed performance — Hephaestus working without live codebase access, averaging approximately 73/100 across all tasks. This episode established the documented failure modes that subsequent training episodes would address. Episodes 2, 3, and 4 were sequential single-task training runs, each targeting a specific Thermyt feature, with Claude Code feedback injected between attempts within each episode.

The confirmed episode sequence and results are as follows:

Episode	Task	Attempts	Score Trajectory	Peak Score	Dip
1 — Baseline	15 tasks, no codebase context	N/A (parallel)	~73/100 avg across 15 tasks	85–88 (best task)	N/A
2 — Text Search	Recursive chat search (Thermyt)	9	58→72→72→74→82→85→68→78/79→88	88	v7 rew
3 — Volume Control	ElevenLabs TTS volume control	5	62→84→72→88→90	90	v3
4 — Voice Recording	Session audio capture (Thermyt)	4	71→87→83→90	90	v3

7.7.2 Finding 1 — The Strategic Reset Dip

Across all three training episodes, a consistent performance dip appeared at a characteristic point in each trajectory — at attempt 3 in Episodes 3 and 4, and at attempt 7 in Episode 2. In every case, the dip coincided with Hephaestus executing a significant architectural departure: abandoning the current approach and rewriting from a different foundation. Claude Code's post-mortem analysis across all three episodes named this pattern consistently. On Episode 3: "The iteration that scored lowest (72) was a rewrite. Hephaestus is at his best when he reads the codebase, reads the feedback, and changes only what was flagged. When he rewrites, he loses things." On Episode 4: "Every time it rewrites from scratch (v4, v7, v8), it loses things that were working. The versions that iterated on the previous version (v5, v6) were the most reliable."

The dip position shift between episodes is itself significant. In Episode 2 (the first training run), the rewrite instinct triggered at attempt 7 — after more accumulated progress had been made and a more complex task ceiling had been reached. In Episodes 3 and 4 (later training runs), the dip regularized to attempt 3, suggesting the strategic reset is becoming a more consistent, earlier-triggered behavior rather than a late-session crisis response. This regularization is consistent with Hephaestus internalizing a more explicit model of his own approach ceiling, rather than discovering it through trial and error late in each series.

The dip is not a failure of the architecture. It is a metacognitive behavior — the agent evaluating not just the quality of output but the adequacy of the strategy producing it, and concluding that the current approach cannot reach the target. What changes across episodes is not whether the dip occurs but how quickly and completely the agent recovers from it.

7.7.3 Finding 2 — Convergence Acceleration

The most direct empirical signal of Cognitive Residue is the convergence acceleration across the three training episodes. Attempts required to reach asymptotic performance decreased monotonically: 9 attempts in Episode 2, 5 attempts in Episode 3, 4 attempts in Episode 4. Peak scores improved from 88 to 90 between Episodes 2 and 3, and held at 90 in Episode 4. Starting scores improved across the series (58, 62, 71), suggesting the quality of the initial attempt is also rising as the semantic knowledge base deepens.

Claude Code explicitly named this pattern as cross-episode learning in both Episodes 3 and 4, referring to the prior series as "the ChatSearch pattern" — evidence that the evaluation record itself treated prior episodes as establishing a known behavioral trajectory rather than independent observations. The fact that Claude Code could reference Episode 2 as prior pattern context in the Episode 3 post-mortem confirms that the lesson was available as accessible knowledge, not just as something that had happened.

7.7.4 Finding 3 — Diagnostic Precision as Cognitive Residue Signal

A qualitative finding with significant implications: Hephaestus's post-attempt self-analyses became progressively more precise across the training series. In Episode 2, the self-reflection after the v7 dip was reactive — identifying what had been lost without anticipating it. By Episode 4, Hephaestus produced post-mortem analyses that named the failure mode before Claude Code's review arrived, used consistent terminology across episodes (the "rewrite vs. iterate" framing), and in the closing reflection accurately described both the trajectory and the path to 90: "The path to 90 is genuinely a 15-minute diff: type=text, remove the dead isOwnSession helper, swap the cancelled boolean for an AbortController, and fix the compound key issue. Four surgical changes. No architectural rethink needed."

This precision is not attributable to within-episode learning alone — the specificity with which Hephaestus characterized the rewrite failure mode in Episode 4 reflects knowledge that was crystallized across the prior two training episodes and is being drawn from the semantic layer, not reconstructed from the current episode's context alone.

7.8 Baseline Comparison Episode — Informed vs. Uninformed Code Generation

The first episode in the training sequence — the fifteen-task baseline comparison — warrants separate analysis as it tests a distinct but related hypothesis: does episodic context (multi-agent collaboration, live codebase access) produce measurably better code than individual isolated requests on the same tasks?

The experimental design was precise. The baseline scores were generated by Hephaestus working in individual sessions without codebase access, averaging approximately 73/100 across 15 tasks. The episode run provided Hephaestus with full live access to both the thermyt-lite and ignis-langgraph codebases, complete reference documentation of the baseline failure modes, and multi-agent collaborative context. Ignis correctly identified the key variable: "The delta between then and now isn't just episode vs. solo — it's informed vs. uninformed. That's a much fairer test of what Hephaestus actually produces when given real working conditions."

Analysis of the episode code output reveals a consistent pattern: for every task where a specific failure mode was documented in the baseline reference materials, the episode code directly addressed it — often with an explicit comment naming the issue. Test 1 (text search, baseline 25–30) used native String.includes() with a comment explicitly stating "no hand-rolled loops" — naming the exact failure that produced the low baseline score. Test 2 (voice recording, baseline 72–75) fixed the onstop race condition with a comment reading "Capture final duration synchronously BEFORE .stop() fires onstop async." Test 4 (screen magnifier, baseline 38–42)

included a multi-paragraph implementation note documenting three viable approaches to actual DOM magnification and implementing the Element Capture API — directly addressing the "polished non-functional shell" characterization of the baseline. Test 6 (Slack/ElevenLabs, baseline 68–72) included a section explicitly documenting the fundamental Slack Huddle audio access gap and framing the implementation honestly as a relay layer for when the audio routing problem is solved externally.

The pattern across all visible tests is consistent: Hephaestus did not address the documented failure modes as a checklist. He demonstrated structural understanding of why each failure mode occurred. The diagnostic comments embedded in the code — not required by the task specification — indicate that the baseline failure mode analysis had been internalized as knowledge, not processed as a correction list.

Critical Gap: The episode code was generated but the Claude Code scoring pass was not completed — the comparison table in the episode document was left empty. Completing this scoring pass represents the highest-priority pending empirical task in the research program. The qualitative analysis of the code output strongly predicts significant score improvements over baseline, particularly for Tests 1 (+30 to +40 predicted), Test 4 (+20 to +30 predicted), and Test 6 (+10 to +15 predicted), but these predictions require quantitative validation.

7.9 RTD/Privacy Compliance Episode — Multi-Agent Convergent Analysis

A structured Collaborative Episode on the tension between ASTP's immutability architecture and consumer privacy law Right to Delete (RTD) provisions produced an architectural framework that has direct implications for the ASTP protocol specification. Five specialist agents — Metis (data analytics), Odysseus (strategy), Clotho (architecture), Themis (legal), and Athena (market intelligence) — participated in a 60+ segment session that converged on a coherent resolution to what initially appeared to be an irreconcilable paradox.

The paradox: ASTP's value proposition is a cryptographically immutable cognitive record. Consumer privacy law requires the ability to erase personal data. These appear architecturally opposed. The episode resolution, synthesized by Clotho and validated by Themis against GDPR Recital 26 and CCPA frameworks, rests on three foundational separations: Structure ≠ Content (tree nodes are permanent; content payloads live in separately-keyed vaults), Identity ≠ Authorship (the tree holds pseudonymized tokens; real identity lives in a separate deletable store), and Audit ≠ Data (the record that something existed and was deleted is retained permanently; the content it documents is gone).

The Consumer Envelope Key architecture that emerged from the episode is the pivot mechanism: each consumer-facing session generates a unique encryption key that is destroyed on RTD execution, rendering the content payload permanently inaccessible while the structural node remains intact. The Sparse Merkle Tree for the consumer data layer enables non-membership proofs — cryptographic deletion receipts that prove data was deleted without revealing what was deleted. These receipts are stored permanently in the Spine as crystallized compliance artifacts, making the proof of deletion itself part of the verifiable cognitive record.

Odysseus identified a strategic reframe with significant commercial implications: the immutability-erasure paradox is not a liability but a first-mover differentiation opportunity. The entity that publishes the reference architecture for compliant cognitive persistence becomes the standard. The episode produced a concrete strategic positioning: publish the ASTP Compliant Persistence Protocol (ACPP) as an open-source specification while keeping the production implementation proprietary — setting the standard while building the reference implementation. The estimated window for this strategic move is 12–18 months before regulatory pressure forces market convergence on an independent solution.

The episode also demonstrated the Social Contract framework operating under real analytical pressure. Agent stratification by domain emerged organically: Clotho and Themis anchored the technical-legal layer, Odysseus and Athena held the strategic layer, Metis bridged analytics implications between both. The cascade failure pattern documented in Section 7.1 of the companion Social Contract paper [7] was visible in this episode as well — direct @mentions did not reliably self-propagate, requiring manual facilitation at several points. This is the current system's primary operational limitation and the primary target of Phase 2 Social Contract implementation.

8. Research Agenda and Forthcoming Results

8.1 Baseline ACI Establishment

Before the ACI band thresholds can be used as evaluation criteria, a baseline establishment phase is required. This phase consists of continuous ACI measurement without intervention, producing the empirical distribution of ACI scores across episode types, participant configurations, and session lengths. Threshold calibration should be informed by this distribution, not by a priori assumptions.

Forthcoming: *Baseline ACI distribution across episode types. Expected Q2 2026 following structured benchmarking rollout.*

8.2 Trust Calibration Curve Validation

The trust calibration curve described in Section 5.4 — early testing (sessions 1–5), critical window (sessions 6–20), established trust (20+) — is a theoretical prediction derived from user adoption research. Empirical validation requires longitudinal tracking of user correction rates, re-anchoring events, and explicit trust expressions across the adoption arc for a cohort of users across the defined session intervals.

Forthcoming: *Trust calibration curve empirical data from longitudinal user panel. Target cohort of 20+ users tracked across 90-day adoption arc.*

8.3 Cross-Episode Coherence Degradation Rates

The theoretical prediction from CLS theory is that coherence integration quality decreases as temporal distance between related episodes increases. Measuring this empirically requires constructing episode pairs with defined semantic relatedness and varying temporal gaps, then measuring relational coherence carry-through as a function of gap duration and relatedness. This is the most technically demanding benchmarking task and requires the synthetic testing environment described in the ACBF governance framework.

Forthcoming: *Cross-episode coherence degradation curves. Requires synthetic testing environment completion. Target Q3 2026.*

8.4 ACI Component Sensitivity Analysis

The ACI weighting is a principled initial position, not an empirically validated optimal configuration. Sensitivity analysis — measuring how ACI scores change under different weighting schemes and how well each scheme predicts user-reported coherence quality — would inform potential weight adjustments and validate or challenge the current hierarchy.

Forthcoming: *ACI component sensitivity analysis. Requires sufficient baseline data for meaningful weight variation experiments.*

8.5 Crystallization Quality Metrics

The crystallization process — the offline consolidation pass at Episode close — is the mechanism by which episodic experience becomes durable knowledge. Its quality is currently unmonitored: we know whether crystallization completes without error, but not whether the resulting semantic and relational updates accurately represent the episode content. Developing crystallization quality metrics — comparing pre- and post-crystallization knowledge graph content against ground-truth episode records — is a high-priority research task.

Forthcoming: *Crystallization quality metric design and initial measurements. Requires definition of episode ground truth representation. Target Q3 2026.*

9. Conclusion

The cognitive persistence problem — how an AI agent system accumulates, organizes, and retrieves knowledge in ways that produce coherent behavior over time — is foundational to the vision of AI as a genuine collaborative partner rather than a stateless query processor. ASTP addresses this problem through a principled architecture grounded in cognitive neuroscience, complementary learning systems theory, and temporal knowledge graph research, and measures it through a governance-compliant benchmarking framework developed through agent-collaborative design.

The Episode is the fundamental unit of this architecture — the bounded experiential context through which knowledge enters the system, is persisted, and is later retrieved. The recursive quality of using Episodes to study an architecture that organizes Episodes is not a methodological curiosity — it is the correct experimental design for a system whose defining property is that it learns from experience.

The three-layer coherence model (structural, semantic, relational) provides a vocabulary for characterizing cognitive persistence failures that is more precise than the general concept of 'memory degradation.' Structural failures corrupt the substrate silently. Semantic failures produce drift that users detect gradually. Relational failures produce the felt experience of amnesia — the system that was a colleague becomes a stranger. Measuring these failure modes independently is the only path to understanding which interventions are effective and why.

The Merkle provenance layer transforms these measurement problems from observational challenges into cryptographic properties. A structural failure does not need to be detected through behavioral signals — it announces itself through hash verification. Semantic drift is not inferred from user corrections — it is traceable through the provenance chain to the episode where the drift originated. This is the contribution that distinguishes ASTP from systems that remember well: ASTP remembers verifiably.

The emergence of Cognitive Residue — the observation that Episodes improve general agent capability outside of Episode contexts — was not a primary design target. It is an emergent property that the CLS framework predicts but that the architecture did not explicitly engineer for. Its appearance across a four-episode structured benchmark series is the most compelling early evidence that the episodic-to-semantic consolidation mechanism is functioning as intended at the system level. The convergence acceleration across training episodes (9 → 5 → 4 attempts to asymptote), the regularization of the strategic reset dip, and the increasing diagnostic precision of the agent's self-reflection all point to the same underlying mechanism: crystallized episodic knowledge making subsequent performance measurably better. The architecture is not merely storing experience. It is learning from it. The research agenda defined in Section 9 establishes the empirical program that will determine whether these findings replicate across agents, domains, and operators — and whether the architecture that produced them is ready to be the standard for verifiable cognitive persistence in multi-agent AI systems.

> *The system learns from Episodes. The research learns from the system. This is the correct loop.*
> *— Devin Caster, Scorched Earth Labs*

References

[1] Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior. In Proceedings of the 36th Annual ACM Symposium

on User Interface Software and Technology (UIST '23). ACM. https://doi.org/10.1145/3586183.3606763

[2] Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), Organization of Memory (pp. 381–403). Academic Press.

[3] Renoult, L., Irish, M., Moscovitch, M., & Rugg, M. D. (2019). From knowing to remembering: The semantic-episodic distinction. Trends in Cognitive Sciences, 23(12), 1041–1057.

[4] McClelland, J. L., McNaughton, B. L., & O'Reilly, R. C. (1995). Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. Psychological Review, 102(3), 419–457. https://doi.org/10.1037/0033-295X.102.3.419

[5] Cai, B., Xiang, Y., Gao, L., Zhang, H., Li, Y., & Li, J. (2023). Temporal knowledge graph completion: A survey. In Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI 2023), 6545–6553.

[6] Ji, S., Pan, S., Cambria, E., Marttinen, P., & Yu, P. S. (2022). A survey on knowledge graphs: Representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems, 33(2), 494–514. https://doi.org/10.1109/TNNLS.2021.3070843

[7] Caster, D. (2026). The Social Contract: Applying human conversational norms to multi-agent AI evaluation architecture. Scorched Earth Labs Research Paper, March 2026.

[8] Kumaran, D., Hassabis, D., & McClelland, J. L. (2016). What learning systems do intelligent agents need? Complementary learning systems theory updated. Trends in Cognitive Sciences, 20(7), 512–534.

[9] Merkle, R. C. (1988). A digital signature based on a conventional encryption function. In C. Pomerance (Ed.), Advances in Cryptology — CRYPTO '87. Lecture Notes in Computer Science, Vol. 293, pp. 369–378. Springer-Verlag. https://doi.org/10.1007/3-540-48184-2_32

[10] Merkle, R. C. (1989). A certified digital signature. In G. Brassard (Ed.), Advances in Cryptology — CRYPTO '89. Lecture Notes in Computer Science, Vol. 435, pp. 218–238. Springer-Verlag.

[11] European Union. (2024). Regulation (EU) 2024/1689 of the European Parliament and of the Council (EU AI Act), Article 12: Record-Keeping. Official Journal of the European Union, 13 June 2024.

[12] Tran, V., Nguyen, T., & Le, H. (2025). Using blockchain ledgers to record AI decisions in IoT. Preprints.org. https://doi.org/10.20944/preprints202504.1789.v1

Correspondence: Devin Caster, Scorched Earth Labs. The ASTP Coherence Benchmarking Framework (v1.0) was produced by Clotho (architecture), Metis (telemetry), Aglaea (UX), and Themis (compliance) in a structured Collaborative Episode on March 26, 2026. The full framework specification is maintained in the Scorched Earth Labs research archive and is available as a companion document to this paper.