RESEARCH PREPRINT · MAY 2026

Architecture of Work Continuity in Human–AI Systems

Authority, Persistence, and Semantic Handoff

Author	Gh. Rotaru (Giorgio Roth)
Affiliation	ContinuumPort Initiative · continuumport.com
Version	v1.1
Date	May 2026
Status	Preprint · This document is an evolving research framework and may evolve.
References	JAIR Submission #22885 · osf.io/b8sgr

Abstract

Human–AI collaboration increasingly involves long-running tasks such as software engineering, research, and system design. However, continuity of work across sessions remains poorly defined. Existing systems typically rely on conversational persistence, context replay, or implicit model memory—mechanisms that conflate continuity of work with continuity of agent presence.

This paper proposes a unified architectural framework for work continuity in human–AI systems. The framework integrates two complementary components: (1) a structural model of persistent AI systems defined by the system state $\Sigma = (D, A, Auth)$ representing declarative state, adaptive memory, and execution authority; and (2) a normative semantic handoff mechanism implemented through directional continuity and the CP-Core capsule structure, which preserves only task-relevant semantic state at session boundaries.

> *"The architecture separates system persistence from work transfer by introducing the concept of a semantic freeze point—at which interaction terminates and a minimal transferable work state is emitted."*

The architecture separates system persistence from work transfer by introducing the concept of a semantic freeze point, at which interaction terminates and a minimal transferable work state is emitted. By constraining what may persist across sessions, the framework enables deterministic continuation of work across heterogeneous AI systems without requiring persistent agent identity, conversational replay, or implicit memory.

1 Introduction

Human–AI interaction increasingly supports complex work that unfolds across multiple sessions. Users expect tasks initiated in one interaction to continue in subsequent sessions, often across different systems or models.

Current approaches attempt to preserve continuity through mechanisms such as retained conversational history, extended context windows, implicit model memory, and persistent agent identity. While these techniques improve short-term usability, they introduce structural weaknesses when work must be resumed, audited, or transferred.

Most importantly, these approaches conflate two distinct forms of continuity:

- continuity of work
- continuity of agent presence

This paper argues that these concepts must be separated. Work continuity should be grounded in explicit semantic state, not in the persistence of conversational interaction or simulated identity. To formalize this separation, we introduce a unified architecture consisting of a system-level persistence model and a task-level semantic handoff mechanism.

2 Persistent AI Systems

A persistent AI system maintains internal state across time. At the architectural level, the state of such a system can be represented as:

$$\Sigma = (D, A, Auth)$$

System state as a triple of declarative state, adaptive memory, and execution authority

D — **Declarative State.**
Explicit knowledge relevant to current tasks, such as goals, decisions, and structured representations of work.

A — **Adaptive Memory.**
Implicit or learned information accumulated through interaction or training.

$Auth$ — **Execution Authority.**
Constraints that determine which actions are permitted within the system, including safety policies, resource limits, and governance rules.

This structure captures the operational state of a persistent AI system. However, not all elements of Σ should transfer across session boundaries. The question of what constitutes a minimal and portable subset is central to the framework proposed here.

3 The Continuity Problem

When long-running work spans multiple sessions, several structural failure modes appear. These issues arise because continuity is implemented through interaction persistence rather than explicit work state.

Fragile Context.
Conversation history may be truncated or lost, preventing reliable continuation. What appears to be a continuous interaction is in fact a lossy reconstruction at each session boundary.

Opaque Memory.
Implicit model memory cannot be inspected, audited, or transferred. Decisions appear to be "remembered" by the system rather than explicitly restated and verified.

Responsibility Leakage.
When state is stored implicitly, it becomes unclear which agent—human or AI—is responsible for any given decision in the task. Accountability dissolves into the interaction history.

Identity Reification.
Users begin to treat the AI as a persistent entity rather than as a computational process. This leads to misplaced trust in continuity and misaligned expectations about what the system can and cannot recall.

4 Work vs. Presence

The proposed framework introduces a strict separation between two domains. Presence is intentionally treated as non-transferable by design—ensuring that work continuity does not depend on simulated agent identity.

SEMANTIC WORK — TRANSFERABLE	PRESENCE — NON-TRANSFERABLE
• Objectives	• Identity
• Decisions	• Personality
• Constraints	• Conversational tone
• Unresolved questions	• Emotional context
• Current task state	• Autobiographical memory

5 Minimal Transferable Work State

For work to continue across sessions, a minimal semantic representation must be preserved. We define the transferable work state as:

$$W = (I, S, C, N)$$

$W \subset \Sigma$ — *the transferable state is a projection of the full system state*

I — **Intent.**
The objective of the task—what the work is ultimately trying to achieve.

S — **Semantic State.**
Established decisions and current progress—the accumulated semantic content of the work.

C — **Constraints.**
Boundaries that must not be violated in any continuation of the work.

N — **Next Action.**
The immediate continuation step—the concrete first action a receiving system must execute to resume work deterministically.

6 The Semantic Freeze Point

To separate interaction from work continuity, the framework introduces the concept of a semantic freeze point. This is the formal boundary at which a session terminates and a portable work artifact is generated. At the freeze point:

1. conversational interaction terminates
2. work-relevant semantic content is identified and extracted from the session
3. non-transferable elements (identity, tone, emotional context) are explicitly excluded
4. a CP-Core semantic capsule is emitted—the minimal, portable representation of the work state

This boundary prevents conversational artifacts from contaminating the continuation state. The freeze point is not merely an implementation detail—it is the primary mechanism by which the architecture enforces the separation between work and presence.

7 CP-Core: Semantic Capsule

The ContinuumPort project operationalizes transferable work state through CP-Core, a minimal semantic capsule. CP-Core intentionally performs lossy semantic compression—information that cannot be expressed within this structure is discarded.

Field	Role	Description
`intent`	Task objective	The primary goal — what the work is trying to accomplish, stated declaratively.
`constraints`	Boundaries	A list of conditions that must hold in any continuation — non-negotiable task invariants.
`decisions`	Established semantic state	Choices already made that must be respected downstream. Forms the semantic history of the task.
`state_summary`	Current task state	A concise description of where the task currently stands — the "current position" in work space.
`progress`	Continuation status	Indicates completeness and what remains — enables receiving systems to orient immediately.

```
CP-Core {
  intent :  string
  constraints :  list
  decisions :  list
  state_summary :  string
  progress :  string
}
```

8 Directional Continuity

Traditional continuity systems attempt to reproduce past interaction with high fidelity—a fundamentally brittle approach. The framework proposed here instead enforces directional continuity. Continuation systems must preserve intent, constraints, and established decisions. Implementation details may diverge.

> *"Two systems continuing the same capsule are not required to produce identical outputs—but they must remain within the same constraint envelope."*

This distinction is critical: directional continuity is a semantic guarantee, not a behavioral one. It preserves what matters—the work—while allowing the variation that is inevitable across heterogeneous AI systems.

9 From Continuity to Enforcement

Directional continuity governs what is *transferred* across a boundary. It does not govern what is *permitted to execute* once work resumes. These are separate concerns, and conflating them is a design error: a capsule can faithfully carry intent and constraints, yet a receiving system may still attempt actions that violate those constraints during execution.

ContinuumPort therefore separates two layers by design. CP-Core is the transport layer—a passive artifact that carries semantic state. It executes nothing and enforces nothing. Execution governance is a distinct layer: an enforcement kernel that evaluates each transition against a declared geometry and refuses those that fall outside it. Passing an advisory evaluation does not imply execution; the kernel re-evaluates independently, and enforcement lives at the geometry level or it does not exist at all.

The boundary between the two is strict and stated deliberately. The enforcement kernel guarantees consequences relative to the declared geometry—never the safety of the world that geometry is meant to describe. What must not be violated has to be declared; an undeclared risk is not blocked. Defining the geometry correctly remains the engineer's responsibility. The enforcement layer is documented and validated separately; this whitepaper treats it only insofar as it completes the continuity picture: transport carries the work, enforcement governs its execution.

10 System Persistence and Work Handoff

The full relationship between persistent system state and transferable work state can be represented as a sequence of transformations across a session boundary. Crucially, adaptive memory A is intentionally excluded from the transferable work state. Only declarative state D is projected into W, via the CP-Core capsule. This ensures that implicit knowledge accumulated by one system cannot silently contaminate continuation by another.

$$\Sigma = (D, A, Auth)$$

Persistent System State

$$\Downarrow$$

SEMANTIC FREEZE POINT

Session Boundary

$$\Downarrow$$

$$W = (I, S, C, N)$$

Transferable Work State

$$\Downarrow$$

CP-Core Capsule

Portable Artifact

$$\Downarrow$$

$$\Sigma'$$

Next System Instance

11 Validation

The framework is validated at two levels: behavioral observation of the handoff mechanism, and adversarial testing of the enforcement kernel that governs execution once a capsule is continued.

11.1 Behavioral Validation of Semantic Handoff

The handoff mechanism was evaluated through long-running collaborative interactions involving multiple AI systems across separate sessions. Observed behaviors included:

- automatic termination of conversational continuation at session boundaries
- consistent extraction of structured work state across sessions
- generation of handoff capsules with consistent structure across different systems
- successful continuation of work in fresh sessions from structured capsule artifacts

Identified failure modes. Failures occurred when capsules were underspecified, or when users relied on conversational context instead of explicit state. These failures were detectable and reproducible, satisfying falsifiability requirements for the framework.

11.2 Adversarial Validation of the Enforcement Kernel

Semantic handoff answers what may be transferred across a boundary. It does not, on its own, govern what may execute once work resumes. That is the role of the enforcement kernel, and it is validated not by observation but by an adversarial test suite: **1,922 deterministic checks** that execute in under six seconds and must pass in full for the kernel to be considered compliant.

The suite is organized around a single discipline: a system is correct if and only if it passes the invariants, and every claim below corresponds to tests that fail on known-faulty implementations and pass on the reference kernel. Coverage includes:

- **Core execution invariants** — atomicity on failure (no partial state escapes a rolled-back transition), snapshot isolation (observation cannot mutate governed state), and determinism (identical inputs yield identical execution records).

- **Authority enforcement** — local admissibility does not compose into trajectory-level authority; a sequence of individually permitted transitions cannot reach a globally inadmissible state without the kernel's veto.
- **Adversarial attack batches** — thirteen categories including authority laundering, provenance falsification, admissibility erosion, delegated-authority abuse, hostile observation, and commitment-graph attacks.
- **Concurrent pressure** — consistency under concurrent registration, observation, and multi-domain authority contention. Concurrency is handled by *strict verdict serialization*: a divergence verdict, once recorded, cannot be overwritten by subsequent confirmations, and epistemic state updates are strictly sequential. This is an operational guarantee of consistency under contention, not a formal model of concurrent verification — a distinction the accompanying formal work states explicitly.
- **Provenance and authority genesis (D1/J1)** — authority cannot be manufactured by replay; a mechanism's right to act must trace to a legitimate, non-forgeable origin rather than to the mere fact of prior execution.

What the suite does not claim. The kernel enforces the declared geometry; it does not certify that the declaration matches the physical world. An execution that is harmful but admissible under the declared constraints will still execute. The correctness guaranteed here is relative to the declared state space — never a guarantee about the world that space is meant to describe. Closing that gap remains the responsibility of whoever defines the geometry.

12 Implications for Human–Computer Interaction

The framework suggests a significant shift in HCI design philosophy. Instead of persistent conversational agents, systems can support long-running work through artifact-based continuity. Users interact with systems that execute tasks within a session, emit semantic capsules at session boundaries, and resume work from structured artifacts.

This approach reduces anthropomorphic confusion while increasing accountability—users maintain clear visibility into what state has been captured and what will be transferred. The implications extend beyond technical architecture: artifact-based continuity changes the nature of the human–AI relationship from one where the AI is treated as a persistent collaborator with memory, to one where the AI is a capable executor within a well-defined work context.

13 Conclusion

> *"The key principle is that continuity should be a property of work, not of agents."*

This paper presents a unified framework for work continuity in human–AI systems. By separating persistent system architecture from transferable work state, the framework enables reliable continuation of complex tasks across sessions and systems.

Explicit semantic handoff structures such as CP-Core allow systems to resume work deterministically without requiring persistent identity, conversational memory, or opaque internal state. As AI-assisted collaboration becomes more widespread, such explicit handoff mechanisms may become foundational infrastructure for long-running human–AI workflows.

Related Research

This whitepaper describes the work-continuity architecture. The formal results, threat model, and governance analysis are developed in the accompanying papers.

Under peer review — Journal of Computer Security (JCS-26-0364).
Trajectory Integrity in Persistent AI Systems: Why Local Verification Is Structurally Insufficient. The composition lemma: strictly local authority cannot enforce trajectory-level safety.

SSRN 6533358
— *Execution Control in Persistent AI Systems: A Geometry-Constrained Model.* papers.ssrn.com/sol3/papers.cfm?abstract_id=6533358

SSRN 6271360
— *Structural Bifurcation in Adaptive Systems: A Formal and Governance Analysis of Task and Identity Continuity.* papers.ssrn.com/sol3/papers.cfm?abstract_id=6271360

Preprints — OSF.
Companion papers: 10.17605/OSF.IO/B8SGR, M8YBN, QWF8A, W7Q9N.

Conceptual framework.
AI Architectural Thinking — the 58-chapter development of the persistence, governance, and continuity model underlying this architecture.

A CP-Core Capsule Schema

Machine-readable schema for semantic capsules. This schema represents the normative structure for all CP-Core compliant handoff artifacts.

```
CP-Core {
  "intent" :  string
  "constraints" :  ["list"]
  "decisions" :  ["list"]
  "state_summary" :  string
  "progress" :  string
}
```

B CP-NORM-H01 Specification

Normative definition of the semantic handoff protocol. This specification forms the normative basis for cross-system work continuation in ContinuumPort.

The protocol defines:

Semantic freeze conditions
When interaction terminates and work state is extractable.

Capsule generation rules
Structure and completeness requirements for CP-Core capsules.

Continuation constraints
Requirements for receiving systems to honor capsule content.

Directional consistency requirements
Invariants that must be preserved across all continuation instances.