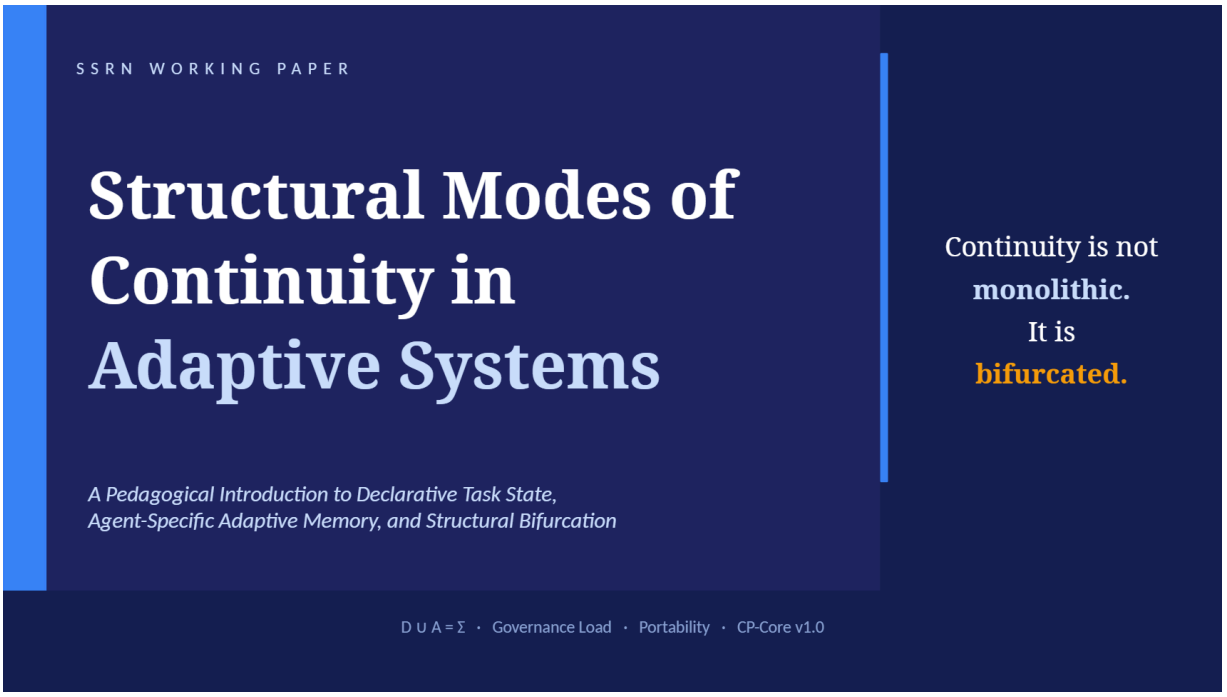

Figure 1: Title slide — SSRN Working Paper overview.

Structural Modes of Continuity in Adaptive Systems

A Pedagogical Introduction to Declarative Task State, Agent-Specific Adaptive Memory, and Structural Bifurcation

SSRN Working Paper | Σ = D ∪ A · Governance Load · Portability · CP-Core v1.0
Gheorghe Rotaru (Giorgio Roth) | ContinuumPort

Section 1: A Familiar Experience

Imagine you use the same AI assistant every day for six months. At first, it feels generic. Over time, it becomes increasingly tailored to you. It remembers your preferred writing style, your coding conventions, your recurring research topics, and the way you phrase questions.

After months, the interaction feels frictionless. Then one day, the system is replaced. The new version is more capable — faster, more accurate. But something feels off. It doesn't 'know you.' Nothing technical is broken. Yet something significant has been lost.

What exactly was lost?

Section 2: Two Interpretations

The intuitive answer: 'It lost memory.' That seems correct. But pause. Was what you lost simply information? Or was it something structurally different? There is a difference between losing a file

and losing a relationship. The technical description might be 'state reset.' But the lived experience feels closer to discontinuity of identity.

→ This suggests: Not all continuity is the same kind of thing.

As illustrated in Figure 2, the intuitive answer ('it lost memory') is technically correct but structurally incomplete — the lived experience points to an unexamined architectural distinction.

Figure 2: The Problem — A Familiar but Unexamined Experience. Intuitive vs. structural framing of continuity loss.

Section 3: Current Approaches and Hidden Costs

Modern AI systems increasingly offer persistent memory features: ChatGPT Memory remembers facts across conversations; Claude Projects maintains context for workstreams; GitHub Copilot adapts to coding patterns. The implicit assumption: More memory → better continuity → better service.

But consider the costs:

- Data Privacy: Every remembered preference is stored data. Who owns it? Where? For how long?
- Platform Lock-in: If the AI 'knows you,' switching providers means losing that investment. • Expectation Dependency: If the AI remembers incorrectly, the relationship feels broken. • Identity Accumulation: Over time, the AI develops a model of you — this is relationship formation, not neutral tool use.

First crack in the assumption: Maybe 'more memory' isn't always better. Maybe we need to distinguish between different types of memory.

Section 4: The Core Question

Before we design systems, we must answer precisely: When we want an AI to 'continue' across sessions, what exactly must persist? Is it the task state? The relationship history? User-specific

adaptations? Conversational memory? This is not a UX question. It is an architectural question. And it turns out there are two fundamentally different answers.

Section 5: A Thought Experiment

Suppose you're working on a software refactoring project with an AI assistant. Scenario A: We remove all conversational history, but preserve a structured record containing: the project goal, constraints, decisions made, and current state. Could a different AI pick up and continue the work? Yes. The project could continue.

Scenario B: We preserve conversational memory but remove the structured project record. Can the task reliably continue? Often, no.

> → This reveals: Continuity of task may not require continuity of relationship. We may have been conflating two distinct phenomena.

Section 6: What We've Discovered

We have observed: (1) Persistent memory creates personalized, relationship-like continuity; (2) Removal of that memory produces relational discontinuity; (3) Task continuation doesn't obviously require identity persistence; (4) Therefore, 'continuity' may not be a single architectural primitive.

The central question becomes: What kinds of continuity should be structurally possible — or impossible — in adaptive systems?

Reader Check-in (Part I):

Q1: What's the difference between 'losing a file' and 'losing a relationship' in the AI context?

Q2: Why might task continuation not require relationship history?

Q3: What assumption about memory are current AI systems making?

Section 7: Two Types of State

In Part I, we asked: What must persist for a task to continue? Now we answer precisely. Imagine you keep two separate notebooks while working on a complex software refactoring project.

Notebook 1 — 'Project Record': Contains the goal, constraints, decisions made, and current state.

Notebook 2 — 'Interaction Journal': Contains user preferences, behavioral patterns, stylistic tendencies, historical conversational context.

These notebooks contain fundamentally different kinds of information. One describes what needs to be done (task geometry). The other describes how we've worked together (relational adaptation). For pedagogical clarity, we introduce two formal symbols:

- **D** — Declarative Task State
- **A** — Agent-Specific Adaptive Memory

Figure 3 illustrates this decomposition formally, showing the structural separation between D and A and their respective properties.

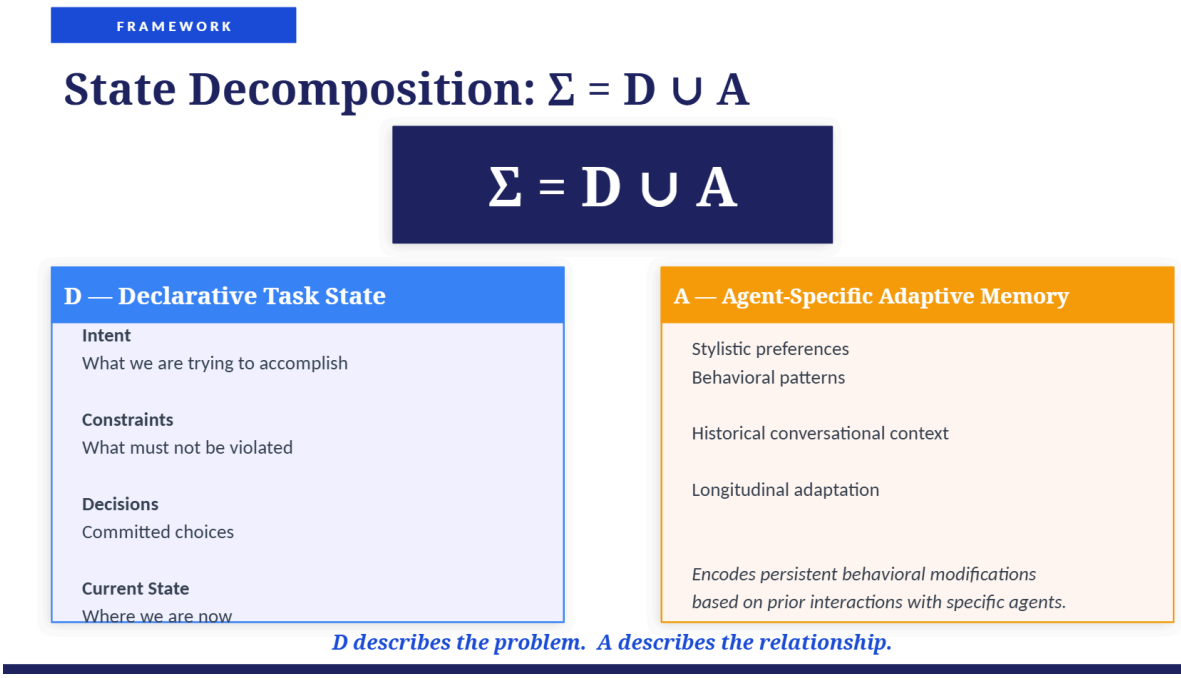

Figure 3: Framework — State Decomposition Σ = D ∪ A. D describes the problem; A describes the relationship.

Section 8: Declarative Task State (D)

D is the equivalent of the 'project notebook.' It contains: Intent (what we're trying to accomplish), Constraints (what we must not violate), Decisions (what choices are already fixed), and Current State (where we are now). Critically, D does not contain narrative history, user-specific adaptations, or how we felt when we decided something. It is oriented toward problem geometry, not toward relationship.

Formal Definition: Declarative Task State (D) is a structured, explicit representation of the conditions necessary for directional task continuation, independent of interaction history.
Key Property — Portability: D is portable. If two different systems receive identical D, they may produce different implementations, but they will preserve direction. This is directional invariance.

Example — Maya's Microservices Migration:

```
Goal: 'Migrate monolithic Flask app to microservices'
Constraints: Zero downtime, support existing API contracts, no DB
schema change
Decisions: Auth service first, API gateway pattern, PostgreSQL + Redis
State: Auth extracted, user service 40% migrated, payment not started
```

Section 9: Agent-Specific Adaptive Memory (A)

A represents everything the system has learned about a specific user or about its relationship with them: stylistic preferences, behavioral patterns, historical conversational context, time-based adjustments. A is not merely data storage — it is longitudinal adaptation.

Formal Definition: Agent-Specific Adaptive Memory (A) is the state component that encodes persistent behavioral modifications based on prior interactions with specific agents.

Critical distinction: D describes the problem. A describes the relationship.

Scenario Comparison — If Maya switches AI mid-project:

- New AI receives D only → Project continues successfully with minor communication adaptation needed
- New AI receives A only → Cannot continue project. Must restart discovery phase. → D is necessary for task continuity. A is sufficient for interaction personalization. But only D is necessary for the work itself to proceed.

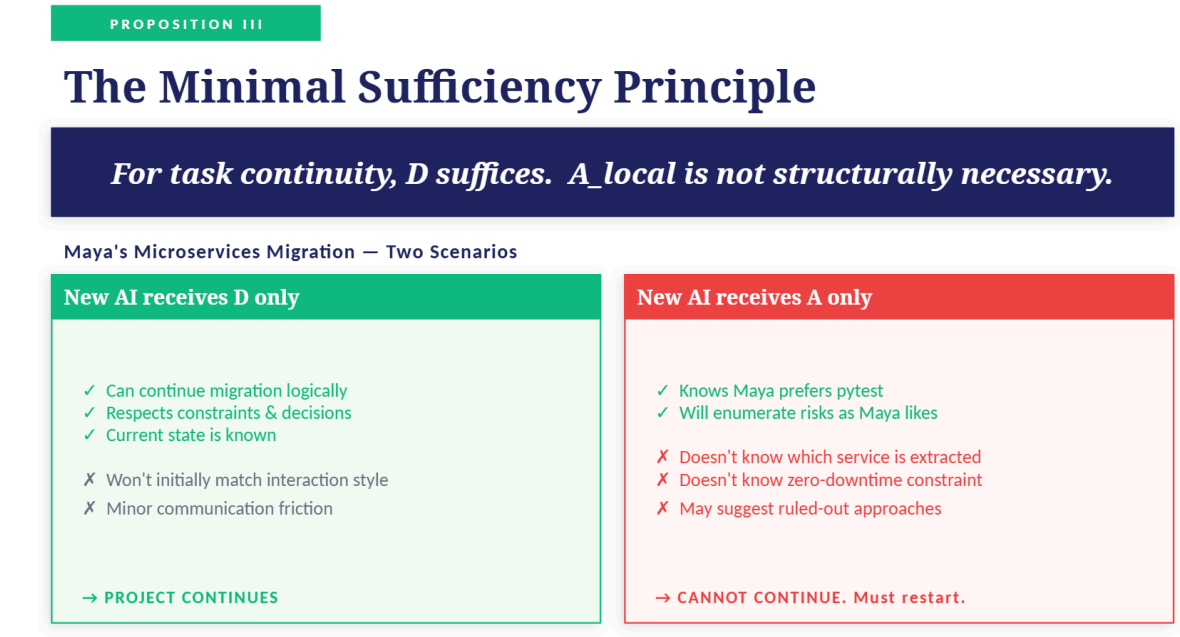

Figure 4: Proposition III — The Minimal Sufficiency Principle. For task continuity, D suffices; A_local is not structurally necessary.

Section 10: The Mechanism of Identity Accretion

Identity accretion does not presuppose consciousness. It does not presuppose intention. It is a structural phenomenon:

- Step 1: System interacts repeatedly with the same agent
- Step 2: Behavioral adaptations are stored in A
- Step 3: These adaptations influence future interactions
- Step 4: Longitudinal behavioral coherence specific to that agent emerges

Consider a thermostat that 'learns' your temperature preferences. After six months it knows you prefer 68°F in the morning and adjusts proactively. Does the thermostat have identity? Philosophically: no. Structurally: it exhibits identity-like structure — behavioral coherence specific to you over time.

Proposition I — Identity Accretion: If A is persistent and grows over time, the system will manifest identity-like structure. This is a structural consequence, not an accidental design choice.

Section 11: Structural Separability

Are D and A inseparable? The surprising answer: No. We can imagine systems where D is preserved and A is zero or has bounded lifetime. These systems can continue tasks but not relationships. This produces a bifurcation.

The Bifurcation Theorem: Continuity splits into two structurally distinct modes — Identity Continuity (preserve A) and Task Continuity (preserve D). These are not extremes on a spectrum. They are categorically different structural regimes.

Current systems conflate D and A: ChatGPT Memory, Claude Projects, GitHub Copilot all store both intertwined. But they don't have to be entangled.

Section 12: Dimensional Reduction & The Minimal Sufficiency Principle

Define total system state: $\Sigma = D \cup A$. If we eliminate A: $\Sigma' = D$. For many task classes, directional execution capacity is not lost — only relational identity is lost. This is dimensional reduction: eliminating a component of state space without destroying the core function of task continuity.

The Minimal Sufficiency Principle: For task continuity, D suffices. A is not structurally necessary. This doesn't mean A has no value — it means A serves interaction optimization, not task continuation.

Scope: This claim is existential and applies only to stabilized task classes where task geometry is externally representable. Operational criteria are outlined in Section 28.

Section 13: Summary of Part II

Part II has introduced the core conceptual architecture of this framework. We have defined D (Declarative Task State) as the portable, agent-independent representation of task geometry, and A (Agent-Specific Adaptive Memory) as the relational, longitudinal, agent-conditioned component. The key structural insight is their separability: $\Sigma = D \cup A$, but D alone suffices for task-directional continuation. The four key implications of D/A separation are: (1) task portability across providers, (2) bounded governance surface when A = 0, (3) elimination of identity accretion in Task Continuity regimes, and (4) structural resistance to lock-in through architectural constraint rather than policy.

Reader Check-in (Part II):

Q1: What's the difference between D (task geometry) and A (relational adaptation)?

Q2: Why does identity accretion not require consciousness?

Q3: Why is eliminating A a 'dimensional reduction' rather than 'functional mutilation'?

Q4: What are the four key implications of D/A separation? (See Section 13.)

Section 14: Governance Load as a Function of Adaptive Memory

In Part II we established $\Sigma = D \cup A$. Now consider what happens when a system maintains A over time: it is no longer just an executor of tasks. It becomes a carrier of longitudinal relationship. This transition is not philosophical — it is structural. And it produces an unavoidable question: When a system 'remembers' specific agents, what obligations emerge?

The Obligation Stack — As |A| grows, obligations accumulate in layers:

• Layer 1 — Consistency: System must remain coherent with its own history.

• Layer 2 — Data Protection: A contains behavioral models of specific people — subject to GDPR, CCPA.

• Layer 3 — Portability: If A is non-transferable, users lose continuity when switching providers. • Layer 4 — Deletion & Rectification: Right to be forgotten, right to correct false memories, right to know what is stored.

Critical observation: These obligations do not arise from policy decisions. They are structural consequences of maintaining persistent A.

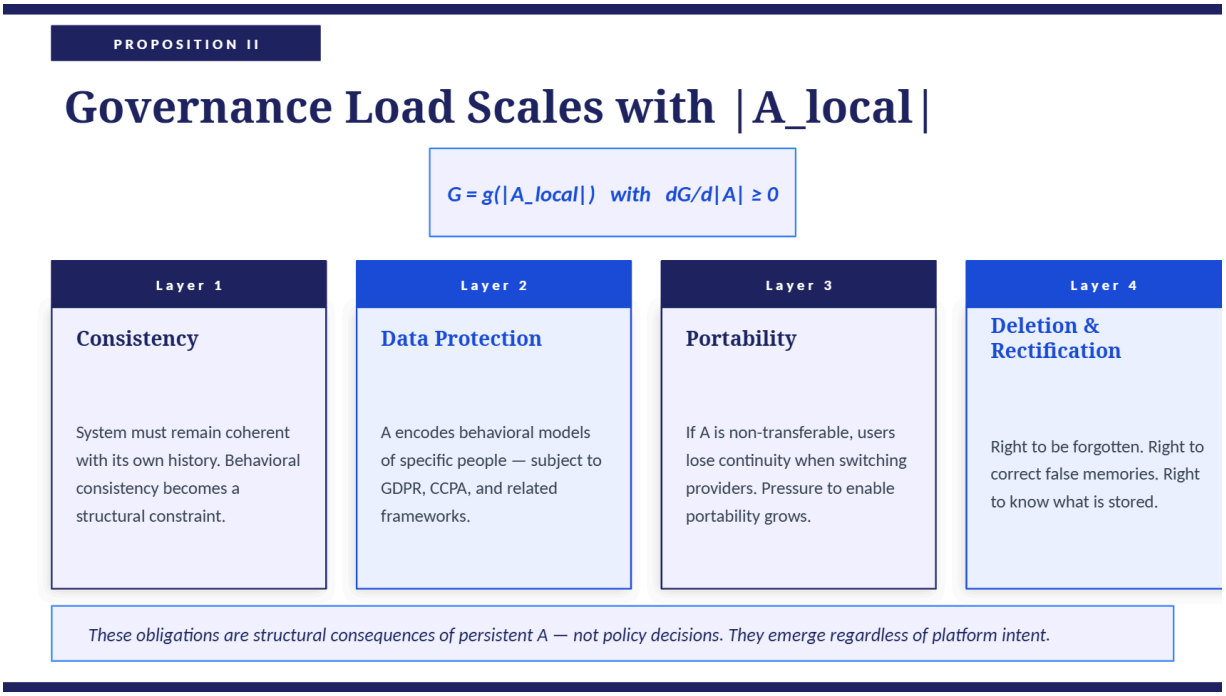

Figure 5: Proposition II — Governance Load Scales with |A_local|. G = g(|A_local|) with dG/d|A| ≥ 0.

Section 14.1: Formal Statement

Governance load G is a function of the magnitude of A_local: G = g(|A_local|). This relationship is monotonically non-decreasing: as A_local grows, governance obligations cannot decrease.

Section 14.2: Structural vs. Policy Origin

The four obligation layers emerge regardless of the platform's stated intentions. A system that accumulates A_local is structurally obligated to manage consistency, protection, portability, and rectification — whether or not its designers acknowledged this in advance.

Section 14.3: Semi-Formal Model of Governance Load

Let |A| = magnitude of agent-specific adaptive memory; G = governance load. Then: G = g(|A|), with dG/d|A| > 0 under standard regulatory and operational assumptions, absent countervailing institutional mechanisms. Governance load increases monotonically with adaptive memory. This is not a moral claim — it is a structural consequence of persistence.

Section 14.4: Centralization Asymmetry

Consider the typical deployment of persistent AI memory: A is stored centrally, A is non-portable, A

is opaque. This creates a structural asymmetry: the user experiences continuity, the platform controls continuity. Switching providers means losing the relationship investment — all adaptations the system made to you over months or years. This is lock-in through identity, not through technical incompatibility.

Section 15: Path Dependence and Identity Lock-In

When A grows over time: system behavior becomes conditioned on interaction history, future responses are shaped by past exchanges, and the system develops expectations — and so does the user. Two systems with identical D but different A will behave differently — even on identical future tasks.

The causal chain: Persistent A → Identity-like structure accumulates → User adapts their behavior to the AI → Mutual co-adaptation deepens → Switching cost increases → Platform asymmetry entrenches.

Section 16: Observation vs. Constraint — Two Paradigms

Most current AI safety approaches follow a reactive paradigm: build, deploy, monitor, detect drift, intervene and correct. Problems: monitoring requirements grow with complexity, harm may occur before detection, cannot observe all failure modes.

The Structural Alternative — Constraint Paradigm: Constrain primitives upstream → reduce deviation surface area → reduce monitoring requirements → governance becomes tractable.

Analogy: Building a children's playground on a cliff requires either a constant watchful guard (observation) or a fence at the edge (constraint). The fence is more reliable because it doesn't require perfect attention. CP-Core is a fence, not a guard.

Section 17: Architectural Strategy — Minimal Sufficiency Applied

If D suffices for task continuation, and A generates governance obligations, path dependence, and monitoring requirements, then a legitimate architectural strategy emerges: Preserve D. Constrain or eliminate A. Where task class permits. This is dimensional reduction as a governance-reducing mechanism.

Reader Check-in (Part III):

Q1: Why does governance load increase as a function of |A|? Name at least three obligation layers.

Q2: What is the difference between 'observation' and 'constraint' as safety paradigms?

Q3: What does it mean for CP-Core to be an 'existence proof'?

Q4: Describe a task class where reducing to D alone would be premature.

Q5: What is the 'stabilization threshold' and why does it matter?

Parts I–III established intuition, structural separation, and systemic consequences. We now introduce a minimal formal model to make the claims precise, falsifiable, and extensible. This is a structural formalization, not a full mathematical theory.

Section 18: Overview of Formal Structure

The formal model consists of five definitions and three propositions, culminating in the Composite Theorem of Structural Bifurcation. Each definition is minimal — it captures only what is necessary for the structural argument.

Section 19: Core Definitions

Definition 1 — Adaptive System

An adaptive system (S) is a system whose internal state evolves as a function of prior interaction. If $\sigma(t+1) = f(\sigma(t), I(t))$, then S is adaptive with respect to update function f.

Definition 2 — State Decomposition

Let Σ denote the full persistent cross-session state of S. We decompose: $\Sigma = D \cup A$, where $A = A_{global} \cup A_{local}$. D is agent-independent by design. A_{global} is population-level conditioning (pretraining). A_{local} is persistent state encoding behavioral adaptation conditioned on interactions with specific agents.

Scope Condition: This framework analyzes A_{local}, not A_{global}. The bifurcation is determined solely by the presence or absence of A_{local}.

Definition 3 — Task-Directional Sufficiency

D is task-directionally sufficient if there exists a policy π such that execution trajectory $\pi(D)$ continues the task without reference to A_{local}. This is an existential condition — it claims possibility of continuation, not optimality, not uniqueness, not universality.

Definition 4 — Identity-Like Structure

System S exhibits identity-like structure toward agent u if the probability distribution of behavior B(t) conditioned on $history_u$ with $A_{local} \neq 0$ differs from B(t) conditioned on D with $A_{local} = 0$, for sufficiently large t. Identity-like structure is binary: $A_{local} = 0 \rightarrow$ no agent-specific conditioning; $A_{local} \neq 0 \rightarrow$ agent-conditioned influence exists.

Definition 5 — Governance Load

G(S) = governance load = the structured set of operational, legal, and relational obligations induced by persistent A_{local}. $G = g(|A_{local}|)$, with $dG/d|A_{local}| \geq 0$ (monotonic non-decrease). g is not assumed continuous — governance obligations often arise as discrete triggers (legal thresholds).

Section 20: Propositions

Proposition I — Identity Accretion

If $A_{local} \neq 0$ and persists across sessions, identity-like structure emerges. **Corollary I.1 — Path Dependence:** Behavior at time t depends on trajectory of A_{local}. Two systems with identical D but different A_{local} diverge.

Proposition II — Governance Scaling

Governance load scales with $|A_{local}|$. Persistent A_{local} generates data protection obligations, consistency expectations, portability constraints, and rectification rights: $G = g(|A_{local}|)$.

Proposition III — Task-Directional Sufficiency

If D is task-directionally sufficient, A_local is not structurally necessary for task continuation. Directional validity requires: intent preserved, constraints respected, committed decisions maintained, state recognized.

This claim is existential and scope-bounded, not universal across all task classes. Failure occurs when critical task constraints remain encoded only in A_local and are not externalized into D.

Section 21: Composite Theorem — Structural Bifurcation

Continuity bifurcates into two regimes determined solely by A_local. The two regimes are categorically distinct — not different points on a spectrum.

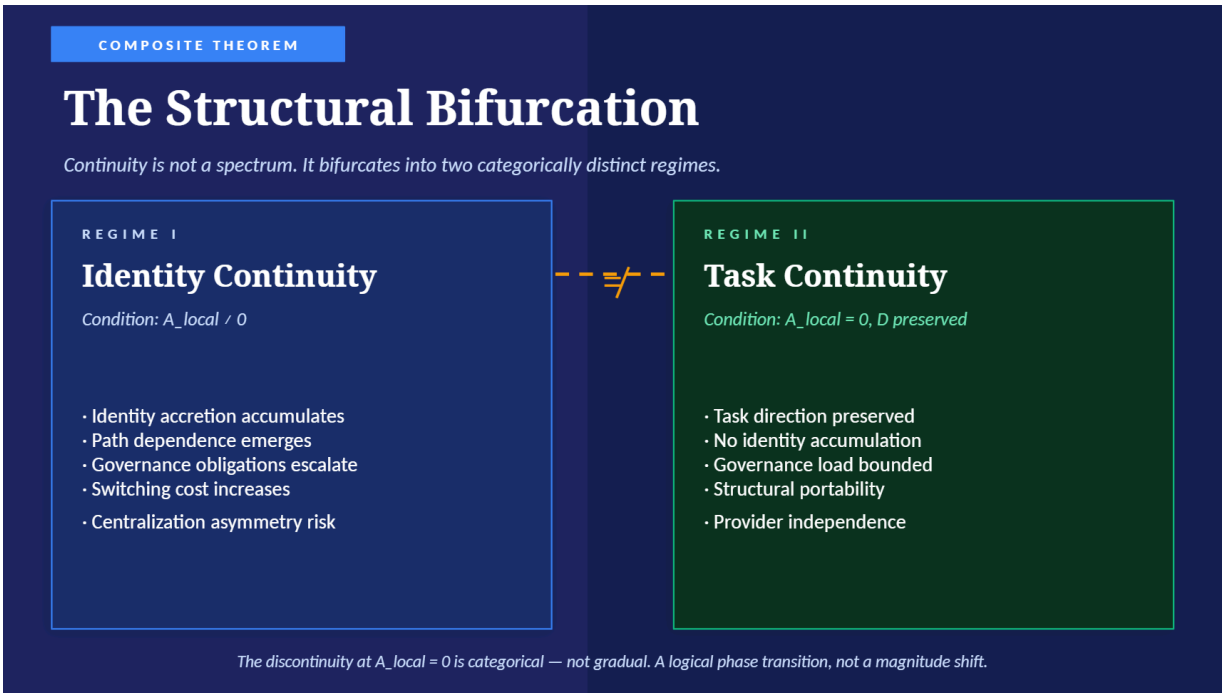

Figure 6: Composite Theorem — The Structural Bifurcation. Continuity is not a spectrum; it bifurcates into two categorically distinct regimes.

Regime I — Identity Continuity (Condition: A_local ≠ 0): Identity accretion, path dependence, governance escalation, switching cost, centralization asymmetry risk.

Regime II — Task Continuity (Condition: A_local = 0, D preserved): Direction preserved, no identity accumulation, bounded governance load, structural portability, provider independence.

Categorical Argument: At A_local = 0, no agent-conditioned influence exists — identity-like structure = FALSE. At A_local > 0, agent-conditioned influence exists — identity-like structure = TRUE. This is a logical phase transition — not a smooth magnitude change. ∴ The bifurcation is categorical.

Section 22: Objections

Can D encode identity? Yes — if D contains agent-conditioned traces, it collapses into Regime I. Separation must be enforced architecturally. Does memory improve quality? Proposition III is existential, not optimality-based. Does distributed A solve governance? Distribution mitigates asymmetry, not structural obligation scaling.

Section 23: Limits of the Model

This model is structural, not measure-theoretic. It does not claim universal applicability, applies above the stabilization threshold, and does not eliminate A_global. The framework makes no claims about performance optimality or quality maximization. It does not claim applicability to exploratory, co-creative, or affectively coupled interaction regimes.

Reader Check-in (Part IV):

Q1: What is the 'stabilization threshold' and why does the framework require it?

Q2: Why is the bifurcation at A_local = 0 categorical and not gradual?

Q3: State Proposition I, II, and III in your own words.

Section 24: Transition from Formal Model to Implications

Having established the formal model in Part IV, we now turn to its implications for system design, governance, and research. The following sections are organized around three axes: architectural regime selection, legal and governance consequences, and open research questions.

Section 25: What the Framework Establishes

- Continuity in adaptive systems is structurally bifurcated — not on a single spectrum. • Identity-like structure emerges from persistent A_local — a structural consequence, not a design choice (Proposition I).
- Governance load scales with |A_local|, not A_global (Proposition II).
- Task-directional continuity does not structurally require A (Proposition III).
- Eliminating A is a structural intervention with categorical consequences.

The central insight is not anti-memory. It is: Different kinds of persistence produce categorically different system properties. This moves the discussion from UX preference to architectural regime selection.

Section 26: Architectural Regime Choice

Regime I — Identity-Centric Systems: A persistent → path dependence, relational continuity, expanding governance surface, vendor entanglement risk, centralization asymmetry tendency. Examples: ChatGPT with Memory, Claude Projects, GitHub Copilot.

Regime II — Task-Centric Systems: A = 0, D preserved → replaceable execution agents, structural portability, bounded governance surface, identity-neutral continuity, no switching cost from relational loss. Examples: CP-Core capsules, stateless API services with explicit state tokens.

These regimes are not 'good vs bad.' They are different commitments with different consequences. Current systems mostly choose Regime I by default — not through careful analysis, but because 'more memory seems better.' This framework argues: the choice deserves scrutiny.

Section 27: Relationship to Adjacent Fields

27.1 AI Safety and Alignment

Current paradigm focuses on behavioral monitoring, value alignment, and capability control. This framework adds: reducing adaptive memory surface (A) is a safety primitive that operates upstream of behavior. Open question: Can alignment be maintained more reliably in Task Continuity regimes?

27.2 Privacy and Data Governance

Established principle: data minimization. This framework extends: minimize not only stored data, but identity-forming adaptive state. Privacy may require architectural restraint on what types of state persist, not just encryption of what does persist.

27.3 Distributed Systems and Stateless Architectures

Known principle: stateless architectures scale better. This framework's analogy: task-centric continuity (A = 0, D explicit) is structurally analogous to stateless service design with explicit session tokens. AI systems have been moving away from statelessness — this framework questions whether that direction is universally optimal.

27.4 Human-Computer Interaction

Dominant framing: personalization improves UX. This framework introduces a counter-dimension: adaptation creates structural obligations and lock-in risk. New research question: What level of personalization is optimal under governance constraints? Is there a Pareto frontier between interaction smoothness and governance tractability?

27.5: CP-Core v1.0 — Concrete Implementation

ContinuumPort CP-Core v1.0 is a concrete implementation of the D/A decoupling strategy. It preserves: Intent, Constraints, Decisions, State summary. It deliberately excludes: Timestamps, Conversation history, User attribution, Stylistic context — each exclusion based on the structural analysis developed above.

CP-Core is not just a format — it is an existence proof: task continuity across AI sessions, providers, and models is achievable with D alone. In the development of this paper, CP-Core capsules were exchanged between Claude and Gemini. Both systems continued work correctly, preserving semantic direction while making different implementation choices.

Available at: github.com/giorgioroth/continuumport-cp-core

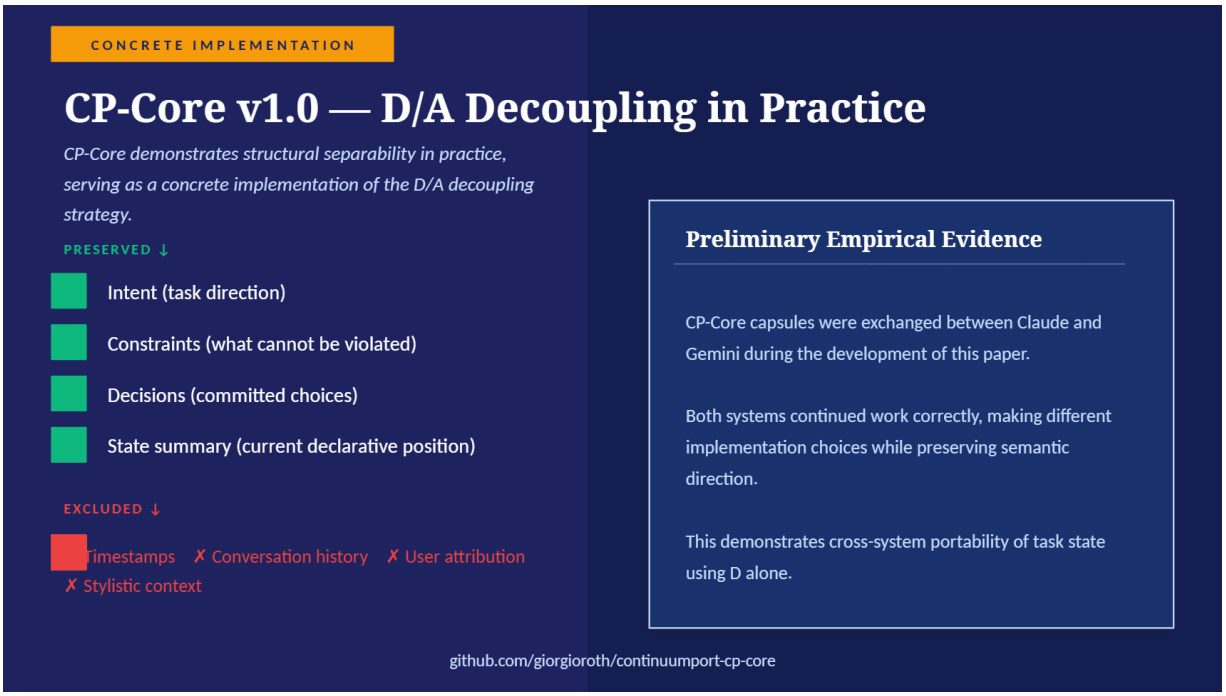

Figure 7: CP-Core v1.0 — D/A Decoupling in Practice. Preliminary empirical evidence from cross-system experiments (Claude ↔ Gemini).

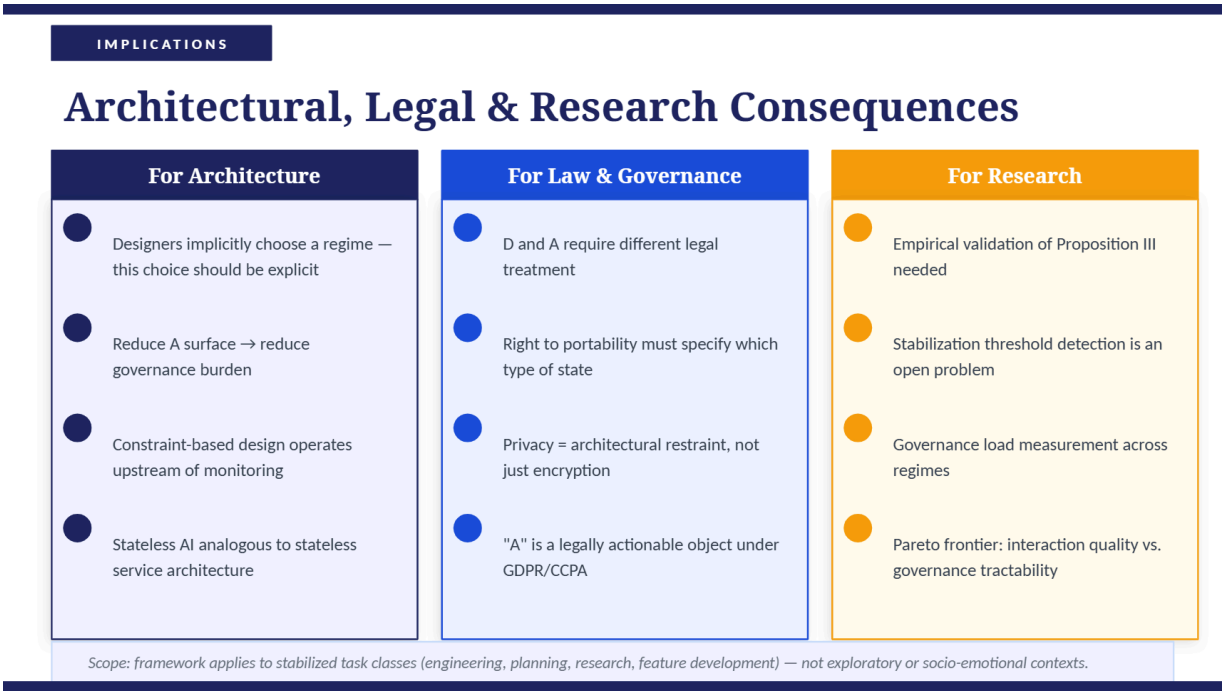

Figure 8: Implications — Architectural, Legal & Research Consequences. Three dimensions: Architecture, Law & Governance, Research.

Section 28: Research Directions

28.1 Empirical Validation of Task-Directional Sufficiency

Proposition III asserts D suffices for task continuation in stabilized task classes. Needed: systematic experiments comparing D-only vs D+A handoffs, measuring directional alignment,

completion success rate, time to completion, and error types.

28.2 Measuring Governance Load

Proposition II asserts G = g(|A|) with dG/d|A| > 0. Proposed metrics: legal compliance overhead, engineering complexity, user support burden, audit surface. Critical question: Does governance load exhibit phase transitions or grow smoothly?

28.3 Detecting the Stabilization Threshold

The framework claims validity for 'stabilized' tasks but provides no algorithmic test. Proposed indicators: decision density, constraint clarity, query convergence, state externalizability. Goal: a reliable threshold detector that makes scope conditions operationalizable.

Section 29: Industrial Implications

- Systems should explicitly declare their continuity regime (Identity vs Task)
- Compliance architectures should treat A as risk-amplifying surface
- Procurement decisions should evaluate governance cost, not just feature richness
- 'Right to portability' may need to specify what type of state is portable

Section 30: Limits of This Framework

This framework does NOT: solve AI alignment, eliminate algorithmic bias, apply universally to exploratory or socio-emotional work, prove quantitative optimality, provide empirical performance guarantees across all adaptive system deployments.

This framework DOES: introduce structural separability of D and A, expose hidden governance gradients, provide a new axis of architectural reasoning, make precise falsifiable claims, offer a constructive alternative via CP-Core.

Section 31: From Regime Selection to Research Program

The framework's most important contribution may be methodological: it reframes the question from 'how much continuity?' to 'which kind of continuity?' This reframing opens a new research program at the intersection of AI architecture, data governance, and safety engineering.

Section 32: Final Synthesis

State decomposition: $\Sigma = D \cup A$. Regime determination: A persistence defines the continuity regime.

In Identity Continuity ($A \neq 0$): Identity-like structure accumulates, governance obligations scale, path dependence increases, lock-in risk emerges.

In Task Continuity ($A = 0$): Task direction preserved, identity accretion collapses categorically, governance remains bounded, structural portability emerges.

The discontinuity at $A_{local} = 0$ is not gradual. It marks the boundary where agent-specific conditioning ceases to exist — not weakens, but disappears. This discontinuity has been invisible because current systems entangle D and A. This paper argues they need not be entangled. Separability is possible.

The question is not 'more continuity or less?'

The question is: 'Which kind?'

And unlike many questions in AI safety, this one has a structural answer.

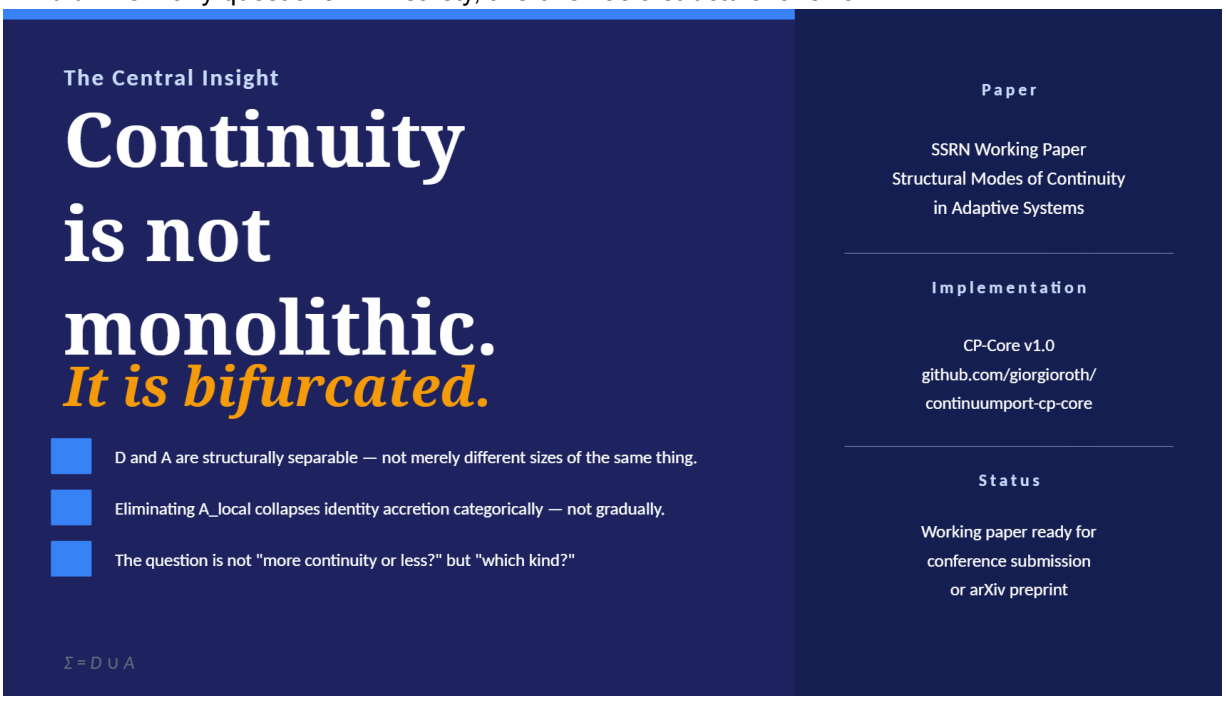

Figure 9: The Central Insight — Continuity is not monolithic. It is bifurcated.

Acknowledgments: This work emerged through adversarial collaboration between multiple AI systems (Claude, Gemini) and iterative human synthesis. CP-Core v1.0 is available at: https://github.com/giorgioroth/continuumport-cp-core