Beyond Chat Memory: Layered Memory Architectures for Persistent Expert Systems

Jordi Buskermolen
April 2026

Paper Type and Audience

This is a **theory / position / systems paper with an architectural proposal**, not primarily an empirical benchmark paper.

Its primary contribution is conceptual: it defines a system category, clarifies the memory problem that category creates, proposes an architectural response, and sketches an evaluation frame better suited to that problem than recall-only metrics. Future implementations, case studies, and experiments may strengthen the argument, but the argument here is fundamentally theoretical.

The intended audience includes researchers working on AI memory and long-lived agents, product and system architects building persistent AI tools, and practitioners designing software around specialized AI roles rather than generic chat assistance.

One-Sentence Thesis

Persistent expert systems require a different memory architecture than generic assistants because their value depends not only on recall, but on preserving identity, scope, temporal truth, and accepted outcomes over time.

Abstract

Recent work on AI memory has concentrated on recall, personalization, and long-context retrieval. These advances have materially improved general assistants, which benefit from remembering user preferences, revisiting past conversations, and grounding responses in external sources. However, they do not fully address the demands of **persistent expert systems**: specialized intelligences that inhabit a durable role, operate within a bounded domain, and build up a history of validated work across evolving projects.

In such systems, memory must do more than retrieve semantically similar fragments. It must help the system remain recognizably the same expert over time, distinguish current truth from historical truth, retain accepted decisions and outcomes, and retrieve according to work scope rather than similarity alone.

This paper argues that persistent expert systems create a distinct **continuity burden**: they must preserve role coherence, current truth, and validated progress simultaneously across extended

use. Prior work has addressed pieces of this problem - through retrieval, long-context management, episodic memory, temporal stores, and governance mechanisms. The contribution here is not to discover those concerns in isolation. Each has precedents. The contribution is to show that they belong together as one integrated design problem, and that treating them as separable enhancements to a recall-oriented base is precisely what causes persistent expert systems to fail. The paper proposes a reference architecture that assembles those concerns accordingly.

The paper proposes a layered memory architecture composed of a canonical identity core, scoped working memory, episodic and outcome memory, temporal fact memory, and deep archive recall. It further argues that durable memory in such systems must be governed through promotion pathways, validity states, supersession, and provenance rather than accumulated indiscriminately.

Finally, the paper suggests that prevailing evaluation approaches, which center on recall benchmarks, are incomplete for this class of system. For persistent experts, relevant evaluation dimensions include identity stability, temporal accuracy, scope precision, decision continuity, provenance fidelity, and contradiction handling. Rather than treating memory as a single retrieval problem, the paper proposes that memory for persistent expert systems should be understood as the operating substrate of continuity and trust.

Contribution Statement

This paper makes four contributions.

1. It defines **persistent expert systems** as a class of AI system organized around a distinctive continuity burden: preserving expert role, current truth, and validated work across time rather than merely improving conversational recall.
2. It identifies core failure modes of existing memory approaches when applied to that burden, including identity drift, scope confusion, stale truth reuse, outcome amnesia, and weak handling of provenance and contradiction.
3. It proposes a layered **reference architecture** that synthesizes identity continuity, scoped retrieval, outcome memory, temporal fact management, and governed promotion into a single architectural response to the continuity burden. Each component has precedents; the contribution is the synthesis and its grounding in a clearly defined design target.
4. It outlines an evaluation framework for persistent experts that goes beyond recall-only benchmarks and aligns more directly with the continuity burden such systems are meant to carry.

1. Introduction

AI memory is often framed as a recall problem: how to retrieve the right prior information at the right moment. This framing has driven substantial progress. Long-context models, retrieval-augmented generation, profile memory, and conversation memory all extend the range of what AI systems can retain and reuse. Within that frame, memory is typically treated as an extension of the context window, a way to recover information that would otherwise fall outside the model's active span.

This framing is appropriate for many assistants whose primary task is to answer or assist better in the current moment. It becomes insufficient, however, when the system is expected to **persist as an expert** across multiple sessions, projects, and revisions to the underlying work. In such systems, the central question is no longer only whether relevant past information can be retrieved. The deeper question is whether the system can remain coherent as the **same expert** over time.

That distinction has architectural consequences. A persistent expert is not simply an assistant with larger memory. It is a different kind of software object: one with a durable role, a bounded domain of work, an evolving project context, and a growing history of validated outputs, revisions, and decisions. It is expected to preserve continuity not only of information, but of orientation. It should know what kind of expert it is, what standards govern its outputs, what truths are currently operative, which decisions have already been accepted, and which parts of the past remain relevant to the work at hand.

When such a system fails, it does not merely "forget" in the ordinary sense. It drifts. It may reuse outdated truths as if they were current, surface material from the wrong work scope, ignore accepted outcomes, or behave like a different expert from one session to the next. These failures suggest that **semantic recall is not the same as expert continuity**. A flat retrieval layer can surface similar content without knowing whether that content belongs to the current work domain, whether it remains valid, or whether it was ever ratified as part of the expert's durable operating context. Full transcript replay preserves history, but history is not yet memory. It includes speculation, abandoned paths, temporary assumptions, and contradictions that should not all exert equal influence on future behavior. Profile memory supports personalization, but persistent experts need more than remembered preferences: they need continuity of role, project truth, and validated work.

The claim of this paper is not that prior work has ignored memory structure, temporal validity, or governance. It has not. The claim is narrower and, we believe, stronger: **these concerns have been addressed separately, as optional enhancements to recall-oriented systems, and that separation is itself the problem**. Once AI is designed to inhabit a durable expert role inside an ongoing body of work, identity continuity, scoped retrieval, temporal truth management, outcome memory, and governed persistence stop being independent improvements and become one integrated architectural requirement. The contribution of this paper is to define that target explicitly - naming it the **continuity burden** of persistent expert systems - and to derive a reference architecture and evaluation frame from it.

This is a synthesis contribution, not a discovery contribution. The components of the proposed architecture have precedents in cognitive science, the retrieval-augmented generation literature, knowledge management, and long-context systems research. What has not been done is to show that they belong together as a single coherent response to a distinct design target, and to derive evaluation criteria that match that target rather than the recall-oriented systems they extend. That is the work this paper attempts.

A note on primitives: the proposed architecture can, in many cases, be realized using existing components - typed embedding stores, versioned document records, namespace-partitioned indexes, promotion queues. The claim is not that novel storage primitives are required. The claim is that **a specific structuring of memory becomes necessary once systems are designed around this target**: one that places identity before history in retrieval order, treats outcomes as first-class objects, enforces validity states on temporal facts, and governs promotion explicitly. Without that structure, systems that appear to satisfy C1–C4 individually will still exhibit the continuity failures described in Section 3.

Accordingly, the paper treats memory for persistent experts as an integrated architectural problem. The proposed response is a layered memory architecture separating a canonical identity core, scoped working memory, episodic and outcome memory, temporal fact memory, and deep archive recall. Just as importantly, the paper argues that durable memory should be governed: not every conversational trace should become persistent truth. Memory in persistent experts must be promoted, statused, revised, and attributable if it is to remain trustworthy over time.

The paper proceeds as follows. Section 2 defines persistent expert systems as a distinct class and situates them relative to adjacent system types. Section 3 identifies the limits of existing memory approaches when applied to that class. Section 4 derives explicit memory requirements. Section 5 proposes a layered architecture. Section 6 addresses promotion, governance, and validity. Section 7 situates the proposal relative to prior work. Section 8 introduces an evaluation frame beyond recall. Section 9 discusses implications, tradeoffs, and open questions. Section 10 concludes.

2. Persistent Expert Systems as a Distinct Category

The argument of this paper depends on treating persistent expert systems as more than a convenient variation of chat interfaces with longer memory. If they are simply general assistants with improved recall, then the memory problem can remain largely within the established frame of retrieval, personalization, and context extension. But if they represent a distinct kind of system, then the architecture of memory may need to change accordingly.

2.1 Definition

We define a **persistent expert system** as an AI system that satisfies all of the following conditions:

- **C1 (Specialized role):** it operates within a bounded area of competence or responsibility rather than across arbitrary open-ended queries.
- **C2 (Temporal persistence):** it is intended to endure across sessions and projects, not just within a single conversational episode.
- **C3 (Evolving context):** it works within a changing environment of projects, priorities, and truths, and is expected to remain aligned with those changes.
- **C4 (Cumulative work):** its outputs form part of an ongoing body of accepted, rejected, revised, or superseded work rather than being disposable turns of dialogue.

When these four conditions hold together, the memory problem changes in a way that justifies a different design target. The system is no longer judged mainly by local helpfulness, but by whether it **remains itself** and builds on prior work appropriately over time.

One might object that C1–C4 are continuous variables rather than binary conditions, and that any sufficiently used assistant accumulates them to some degree. That is true, and the category boundary is not a sharp threshold. The claim is more precisely this: as C1–C4 strengthen together, the continuity failures described in Section 3 become not incidental defects but defining product failures. At that point, the memory architecture must be designed around the continuity burden explicitly. The category designation names that design target, not a sharp empirical partition.

A persistent expert is therefore best understood as a specialized software object with a durable role, a bounded domain of work, and an accumulating history of validated interaction. That coherence includes more than tone or style. It includes continuity of purpose, continuity of standards, continuity of the outputs it is responsible for producing, and continuity of how prior work should shape future work.

A final clarification on terminology: the identity core is an operational concept, not a metaphysical one. When the paper says the system must "remain itself", it means that the expert's governed role configuration - its purpose, standards, constraints, and output contract - should be stable across sessions and resistant to drift. This is a memory design requirement, not a claim about machine selfhood.

2.2 A Worked Example: The Strategy Expert

Consider a persistent strategy expert deployed within a startup. Its domain covers product positioning, pricing logic, competitive framing, and go-to-market planning. Over several months of use, the team has worked through multiple strategy iterations, accepted a current ICP definition, revised their pricing model twice, and adopted a specific positioning angle after rejecting two others.

Each of these events is a memory challenge. The current ICP definition should carry more authority than the prior one - not because the earlier version was wrong at the time, but because it has since been superseded. The accepted positioning angle should inform future messaging work without requiring the team to re-adjudicate it. The rejected positioning approaches should be retrievable for context but should not compete with accepted directions as live options. The revised pricing model should be treated as current, with its predecessor treated as historical. And all of this must remain stable as the expert's role across new sessions regardless of how long the gap between sessions was, who initiates them, or what project surface is being worked on.

A flat retrieval system would struggle here. Semantic search might surface the superseded ICP alongside the current one, with no reliable signal about which is operative. Transcript accumulation would carry rejected positioning discussions forward with no status distinction. Profile memory would capture user preferences but not the validated strategic decisions that should govern future expert behavior. These are not retrieval failures in the narrow sense. They are structural failures in how memory represents historical material, validated decisions, and current operating truth.

What the layered architecture would do differently is traced in detail in Section 6.2.1, which walks a comparable scenario - a compliance expert managing evolving certification facts - through the full write path, from raw trace to promoted durable memory with explicit status and supersession. The strategy expert case follows the same logic: each decision event maps to a promotion pathway, each superseded truth requires a status update in temporal fact memory, each accepted output earns a record in episodic and outcome memory. The point is not merely that the system should store more; it is that it should store differently, in a structure that encodes authority, recency, and outcome status rather than accumulating all content in a flat, undifferentiated space.

2.3 Relation to Adjacent System Types

The distinction becomes clearer when contrasted with familiar AI system types.

System type	**Typical scope**	**Continuity burden**	**Primary memory role**
General assistant	Broad, ad hoc queries	Low: per-session convenience	Recall preferences and recent context

System type	Typical scope	Continuity burden	Primary memory role
Embedded copilot	Single document or tool	Local: within file or task	Maintain local task context
Workflow agent	Action sequences and pipelines	Procedural: state during execution	Track state, results, and failures
Long-context chat UI	Conversation history as context	Conversational continuity	Retrieve relevant past turns
Persistent expert	Bounded domain over many sessions	High: identity, truths, decisions, and work	Sustain expert continuity over time

Persistent experts may overlap with assistants, copilots, and agents, but they are reducible to none of them. A general assistant may remember user preferences and prior conversations, but its memory serves mainly to improve convenience and conversational continuity. A copilot embedded in a document or tool may retain local context for the duration of a task, but it is typically not responsible for carrying a durable role across projects. A workflow agent may track procedural state, but its continuity is often procedural rather than epistemic: it continues a sequence of steps, not a specialized body of judgment.

The key distinction is this: a general assistant mainly answers across topics, while a persistent expert **inhabits a role across time**. Once the system is expected to inhabit a role, memory stops being only a matter of access to prior information and becomes a condition of continuity in the system's operating orientation.

A natural question at this point is whether the identity core - the layer that preserves this role - is simply a well-written static system prompt. It is not, for a structural reason: a static system prompt cannot represent the expert's accumulated validated decisions, current project truths, superseded assumptions, and history of accepted outputs. All of those must participate in shaping future behavior, and they change over time. The identity core is therefore not a fixed instruction but a governed, versioned memory object that interacts with the other layers. Maintaining the expert's role consistency is an ongoing memory management problem, not a one-time prompt-engineering problem.

2.4 Why Treat This as a Category Rather Than a Pattern?

A reasonable objection is that persistent experts may be nothing more than well-engineered long-lived agents with structured memory. This paper takes that objection seriously. The claim is not that persistent experts use wholly different primitives from other advanced AI systems. The claim is that the conjunction of C1–C4 creates a **continuity burden** strong enough to justify a distinct design target.

That burden matters because it changes what counts as success. In many agents, memory improves local task performance, personalization, or procedural completion. In persistent experts, memory must additionally preserve a stable operating role, maintain current truth under revision, and carry validated outcomes forward as constraints on future work. Those demands make identity a first-class memory object, make outcome continuity a first-class requirement, and make governed persistence central rather than optional. In that sense, the category claim is not taxonomic for its own sake. It is a design claim: if a system is expected to satisfy C1–C4 together, then continuity failures cannot be treated as minor retrieval defects. They become defining product failures.

The paper therefore uses **persistent expert systems** not as a branding label for "better agents", but as a way to name a design target whose memory requirements become easier to see once that continuity burden is made explicit.

3. Why Existing Memory Approaches Fall Short

Existing approaches to AI memory have made real progress. Retrieval-augmented generation, long-context prompting, profile memory, and transcript search each solve meaningful parts of the continuity problem. They help systems recover prior information, adapt to users over time, and maintain access to materials that would otherwise disappear beyond the active context window. But they were largely developed around a different center of gravity: the **assistant that needs better recall**. When applied directly to persistent expert systems, characteristic failure modes appear.

3.1 F1: Flat Semantic Retrieval vs. Scoped Relevance

Vector search and related techniques excel at finding semantically related fragments. But **semantic relatedness is not the same as operational relevance**.

A persistent expert often works across multiple domains - positioning, pricing, product strategy, website messaging, research - that may use overlapping language while requiring different forms of recall. A discussion of pricing strategy may resemble a discussion of value messaging, which may in turn resemble a conversation about competitive positioning. Purely semantic retrieval can surface the wrong material for the right words. It recovers something similar without knowing whether that memory belongs to the active scope of work, whether it remains authoritative, or whether it was ever accepted as part of the expert's durable operating context.

In persistent expert systems, the relevant memory is often not the most similar fragment, but the most **appropriately scoped** one.

3.2 F2: Transcript Replay vs. Governed Memory

Another common response is to preserve more history inside the active prompt - by replaying prior transcripts or appending summaries of past turns. This can improve local continuity but risks confusing **history** with **memory**.

History contains speculation, temporary assumptions, false starts, discarded options, and contradictory paths. If raw history is carried forward without stronger structuring and status, salience becomes blurred. Old paths that were never accepted can silently reappear. Tentative ideas can harden into pseudo-truth through repetition. A system that continuously reabsorbs its own ungoverned history may look persistent while actually drifting.

3.3 F3: Preference Memory vs. Professional Continuity

Profile and preference memories capture user traits, recurring choices, favorite formats, or biographical details. This is useful for making systems feel more personal, but it addresses only a narrow slice of what persistent experts need.

Persistent experts must remember **accepted outputs**, **active constraints**, **superseded assumptions**, **ongoing project truths**, and **decisions that shape future behavior**. None of

these are adequately captured by a memory model centered primarily on user preferences. The result can be a system that feels personally familiar while remaining professionally discontinuous.

3.4 F4: Timeless Storage vs. Temporal Validity

Much of the memory needed in ongoing expert work is not static fact but **operational truth**: the current ideal customer profile, the active product direction, the present pricing logic, the latest accepted messaging angle, the live constraints of a project.

These truths change. Yet many memory systems treat stored items as effectively timeless once ingested. Even when retrieval is semantically accurate, it may surface information that is historically true but presently wrong. Persistent experts are especially vulnerable to this failure because the most important truths in their domain are often the ones most likely to evolve.

A memory system that cannot represent temporal validity cannot reliably support ongoing expert work.

3.5 F5: Discussion vs. Decision

In persistent expert systems, not all prior content deserves equal authority. Work that has been accepted, approved, or built upon should matter more than work that was merely explored.

Many memory approaches do not treat **validated outcomes** as first-class objects. They store text, embeddings, or undifferentiated summaries, without preserving the distinction between an explored option and a chosen direction. The system then remembers what was discussed but not what was decided. It may return to rejected paths, ignore accepted constraints, or reopen settled questions without recognizing that it is doing so.

3.6 F6: Opaque State vs. Provenance and Contradiction Handling

Persistent experts also need memory that is inspectable and corrigible. In many current designs, items in long-term memory lack explicit provenance or contradiction handling. It may be impossible to tell whether a remembered item came from an explicit user instruction, a trusted artifact, an earlier speculative suggestion by the system, or repetition over time.

Without provenance, the system's remembered state is hard to debug or correct. Without contradiction handling, conflicting facts may simply coexist until one is surfaced by chance. The system can appear to remember a great deal, but lacks disciplined ways to decide what should still count as current.

These six failure modes share a common root: existing memory approaches optimize for **access** to prior material rather than for **structured continuity** of role, truth, and validated work. Strong recall performance is therefore not enough. A system can retrieve the right fragment and still produce the wrong expert behavior if that fragment belongs to the wrong scope, reflects an

outdated truth, ignores an accepted prior outcome, or conflicts with the expert's own durable role.

4. Memory Requirements for Persistent Experts

The failure modes above motivate explicit requirements. We frame these as conditions **R1–R7** that a memory system should satisfy if it is to support persistent expert systems.

R1. Identity Continuity

A persistent expert must preserve a stable sense of **what kind of expert it is**. This includes its purpose, who it serves, the standards it applies, the types of outputs it is responsible for, and the constraints that define its operating frame. The first requirement of expert memory is not recollection, but self-consistency.

R2. Scope-Aware Recall

Persistent expert work is structured. Memory should be structured accordingly. An expert may operate across distinct work domains - positioning, pricing, research, product strategy, workflow design - whose language overlaps while their operational implications differ. The memory layer must retrieve primarily according to **work scope**, not just similarity.

R3. Temporal Truth Management

Persistent experts operate in environments where important truths change. The memory layer must distinguish at least three temporal statuses:

- **Current:** governing truths and constraints
- **Historical:** still informative but no longer operative
- **Disputed or provisional:** recognized as unresolved or low-confidence

Without temporal status, the system is prone to stale truth reuse and hidden conflict.

R4. Outcome Continuity

For persistent experts, **validated outcomes** must be first-class memory objects. Accepted outputs, approved directions, rejected paths, and decisions that have been built into subsequent work should exert stronger influence than content that was merely discussed. A persistent expert should not only remember what was discussed; it should remember what was decided.

R5. Provenance and Governability

Durable memory should be **attributable**, not merely retrievable. At minimum, the system should be able, in principle, to indicate where a durable memory item came from, how it entered long-term memory, and what gives it authority. Memory that cannot be inspected or corrected is difficult to trust in systems expected to persist over time.

R6. Contradiction Handling and Supersession

Expert memory must support **revision**, not just retention. When new information conflicts with old information, the system must be able to mark older items as historical, replace them with updated current state, or mark the relation as disputed if resolution is unclear.

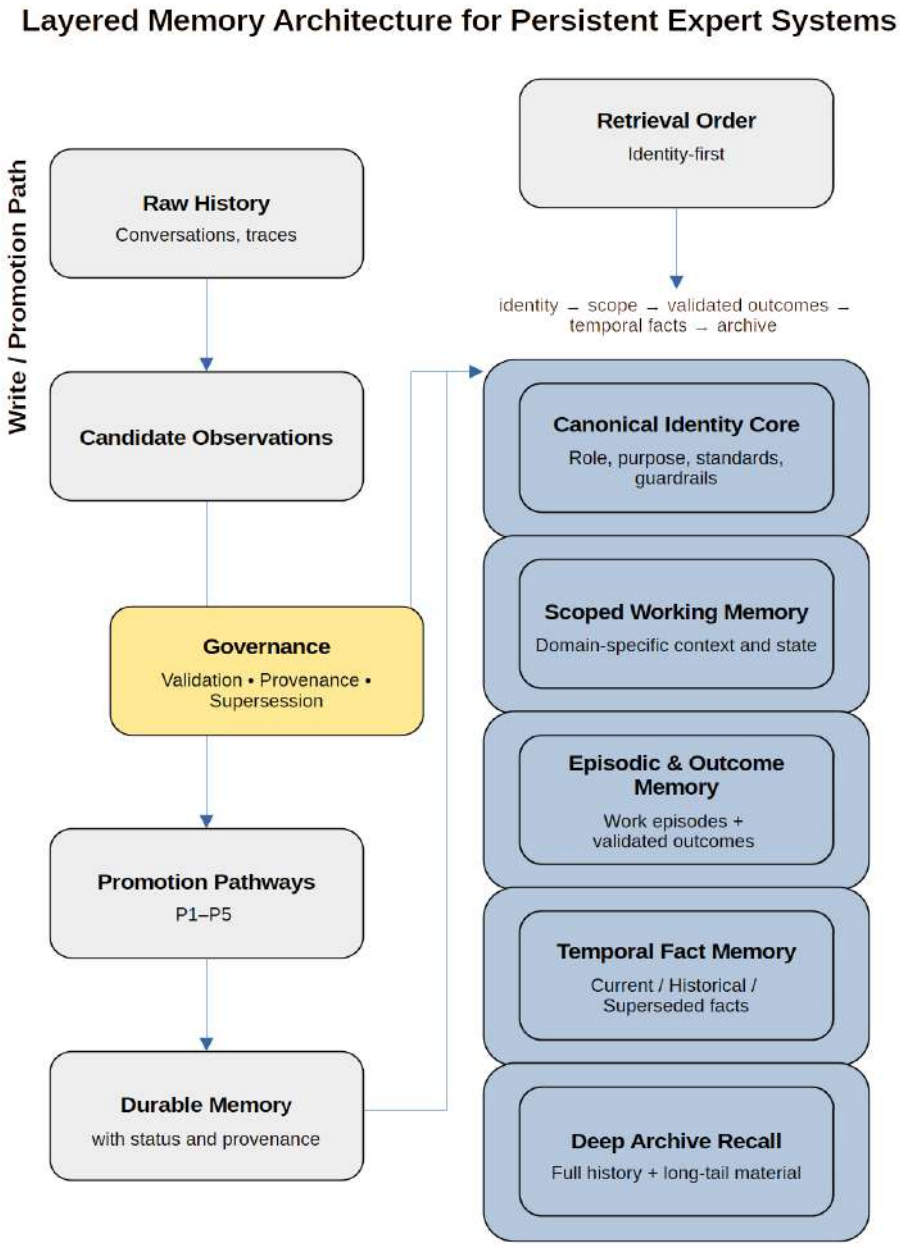
R7. Compression Without Collapse

Persistent experts need long horizons, but cannot carry an ever-expanding transcript into every interaction. The system must be able to compress history into more structured forms - facts, episodes, decisions, current working state - without collapsing important distinctions between current and historical, explored and accepted, inferred and confirmed.

Taken together, R1–R7 imply that memory for persistent experts cannot be implemented as a single undifferentiated store optimized for semantic recall. Different aspects of continuity impose different demands. A plausible response is therefore a **layered architecture** in which distinct forms of memory are stored, promoted, retrieved, and revised according to the role they play in expert continuity.

5. A Layered Memory Architecture

We propose a layered memory architecture designed to satisfy R1–R7 while addressing F1–F6. The architecture separates five layers plus an implied write path. This should be read as a **reference architecture**: a sufficient and well-motivated template for satisfying the requirements of persistent experts, not the only possible realization of them. As noted in Section 1, many implementations of this architecture will use familiar components - typed embedding stores, versioned records, namespace partitions - the contribution being the specified structure and ordering, not novel storage primitives.



5.1 Canonical Identity Core (supports R1)

The **identity core** is the smallest and most stable layer. It encodes the expert's purpose, role, audience, responsibilities, output contract, standing guardrails, and durable operating preferences. Here, "identity" is an operational term: a governed configuration for role, standards, and responsibilities, not a claim about machine selfhood.

This layer should be present on every turn. The system should not have to rediscover its role from prior transcripts. The identity core anchors self-consistency and reduces role drift.

Possible representation: a compact structured document - analogous to a system prompt, but explicitly versioned and governed - storing fields such as `role`, `domain`, `output_standards`, `standing_constraints`, and `durable_preferences`. Unlike a static system prompt, this document is a memory object: it can be updated through explicit promotion (see Section 6), and its prior versions are retained as historical state rather than discarded.

5.2 Scoped Working Memory (supports R2, R7)

Scoped working memory organizes memory according to real work domains rather than a flat recall space. Scopes might include positioning, pricing, product strategy, research, or project-specific workstreams.

The function of this layer is not merely storage, but **ordering**. Scoped structures give retrieval a prior: within this domain, retrieve the most relevant content. This biases the system toward operational relevance rather than lexical resemblance and provides a natural unit for compression.

Possible representation: a namespace-partitioned index, where each work domain contains its own embeddings, summaries, and current state records.

Scopes themselves require maintenance. If scopes are defined once and never revised, retrieval can become brittle as work evolves and domain boundaries shift. If scope definitions shift too freely, the continuity they provide dissolves. Scope maintenance - deciding when to split a scope, merge two related ones, deprecate an inactive one, or reassign items after a project restructuring - is therefore part of the governance problem, not just an implementation detail. Authority over scope definitions should follow the same principles as authority over other canonical truths: legible, corrigible, and updated through explicit pathways rather than silent drift.

5.3 Episodic and Outcome Memory (supports R4, R7)

Episodic and outcome memory captures meaningful work episodes:

- what the system and user were working on
- what materials were consulted
- what conclusions were drawn

- what artifacts were produced
- which outcomes were accepted, rejected, or left open

Accepted episodes - work that has been explicitly adopted, shipped, or integrated - should have stronger influence on future behavior than episodes that remained exploratory. This layer supports continuity of **validated progress** and offers a structured way to compress long histories.

Possible representation: structured episode records with fields such as `scope`, `session_date`, `work_summary`, `outputs_generated`, `outcome_status`, and `linked_artifacts`.

5.4 Temporal Fact Memory (supports R3, R6)

Temporal fact memory stores facts whose validity can change over time. It distinguishes current, historical, and disputed truths.

In persistent expert systems, memory must encode not only what is believed, but **when** it was operative. This layer supports updated strategies, superseded assumptions, evolving constraints, and any other truths whose authority depends on temporal status.

Possible representation: versioned fact records with fields such as `fact_content`, `scope`, `valid_from`, `valid_to` or `superseded_by`, and `validity_status`.

5.5 Deep Archive Recall (supports R7; mitigates F2)

Deep archive recall is the broadest and least privileged layer. It includes full prior conversations, source materials, historical outputs, and the long tail of memory that may still matter in exceptional cases.

The archive remains important, but it should not define the system's active continuity by default. Archive retrieval should extend expert continuity - not silently dominate it.

Possible representation: full-fidelity storage of prior sessions and artifacts, indexed for retrieval but not automatically loaded into active context.

5.6 Retrieval Order as Design

The order in which layers are consulted is as important as the layers themselves. A reasonable retrieval sequence is:

1. Load the **identity core**
2. Load **active scope** and workspace/project state
3. Load relevant **episodic and outcome** memory
4. Apply **temporal fact** constraints and updates

5. Consult **deep archive** only if needed to fill gaps

This sequence biases the system toward continuity grounded in identity and current work before reaching for long-tail recall. Retrieval should proceed from **identity to scope to history**, not from history to identity.

5.7 An Implied Write Path

The architecture also implies a write path. Raw conversational traces should not be written directly into durable memory layers. Instead, they should flow through:

- raw history →
- candidate observations →
- promotion into structured layers (identity, scopes, episodes, temporal facts, archive)

Persistence is thus **earned through promotion**, not granted by default to every conversational trace.

6. Promotion, Governance, and Memory Validity

A layered architecture alone does not guarantee trustworthy memory. The question is not only what can be retrieved, but **what has been allowed to persist, under what status, and with what authority**.

6.1 Raw History, Candidate Observations, Durable Memory

We distinguish three levels:

- **Raw history:** all messages, tool outputs, and intermediate steps produced during the life of the system
- **Candidate observations:** provisional interpretations extracted from history - possible preferences, inferred facts, summarised episodes, likely decisions
- **Durable memory:** items that have been explicitly promoted, assigned status, and integrated into the relevant layer(s)

If every conversational trace hardens directly into durable memory, the system accumulates noise, contradictions, and stale assumptions. Governance is the process of **deciding what crosses the boundary** into durable state and how it is labeled once it does.

6.2 Promotion Pathways

Promotion from candidate observation to durable memory can follow several pathways:

- **P1: Explicit user confirmation** - the user states that a fact, preference, or decision should be treated as durable
- **P2: Approval through workflow** - an artifact or direction is reviewed and accepted in the product's own workflow
- **P3: Repeated stable pattern** - a preference, fact, or constraint recurs across interactions without contradiction, earning higher confidence
- **P4: Trusted structured source** - canonical briefs, reviewed documents, or other authoritative inputs are ingested
- **P5: Outcome confirmation** - a suggestion becomes part of a shipped artifact, a live document, or a decision that subsequent work relies on

These pathways express a simple principle: **persistence should follow validation, not mere mention**.

When promotion pathways conflict - for example, when explicit user confirmation contradicts an inferred stable pattern - the explicit pathway should normally take precedence. Human authority over durable truth should remain legible and final. This creates an unavoidable tradeoff: explicit confirmation reduces drift, but too much confirmation can add friction, while more automated promotion reduces friction but increases the risk of silent memory errors.

6.2.1 A Micro-Scenario: From Raw Trace to Durable State

Consider a compliance expert that helps a team prepare vendor-security responses. During one session, the user says, "For now, ISO 27001 is our primary certification anchor". That statement first enters **raw history**. The system may extract a **candidate observation** that the current certification anchor is ISO 27001. If the user later approves a finalized questionnaire built on that assumption, the fact can be promoted through **P2 (workflow approval)** or **P5 (outcome confirmation)** into **temporal fact memory** as a **current** fact. If, two months later, the team updates its policy and explicitly says, "Use SOC 2 as the primary anchor going forward", that new statement can be promoted through **P1 (explicit confirmation)**, marking the earlier ISO 27001 fact as **historical** rather than merely leaving both facts in retrieval space.

The same session may also generate an accepted answer template. That template belongs not in temporal fact memory but in **episodic and outcome memory**, where it is linked to the relevant scope, outcome status, and source artifact. On a later turn, retrieval should therefore surface the expert's identity core, the active compliance scope, the currently valid certification anchor from temporal fact memory, and the accepted response template from episodic memory before consulting the archive. The point of the example is that continuity depends not only on storing information, but on storing it in the right layer, with the right authority, and with a clear supersession path.

6.3 Status and Supersession

Durable memory items should carry explicit **status**, for example:

- **Current**
- **Historical**
- **Disputed**
- **Archived**
- **Low-trust / inferred**

Status allows the system to reason about memory, not just retrieve it. It is especially important for project truths that change, for decisions that are reversed, and for facts that are later superseded.

Supersession policies determine how conflicts are handled. When new information contradicts old information, the system should:

- prefer appropriately authorized, fresher, or more global facts
- mark superseded items as historical rather than deleting them
- flag unresolved conflicts as disputed

Continuity depends as much on disciplined revision as on durable recall.

6.4 Provenance and Corrigibility

For durable memory, provenance should be preserved where feasible:

- source (user statement, document, tool output, prior conversation)
- promotion pathway (P1–P5)
- relationship to prior items (supersedes, refines, contradicts)
- recency and authority indicators

Provenance supports debugging and correction. If a persistent expert behaves in a surprising way, we want to be able to ask **why does it believe this?** and identify the underlying memory item and its promotion path.

6.5 Human Authority and Automation Balance

Governance does not require that every memory operation be manually supervised. Systems can and should help by suggesting candidate facts, detecting repetition, surfacing possible contradictions, and compressing histories into structured representations.

However, authority over canonical truths - especially identity, high-level constraints, and project-defining decisions - should remain legible and corrigible. Scope definitions themselves also require maintenance: if scopes are never revised, retrieval can become brittle; if they shift too freely, continuity dissolves. When a project restructures, when a domain boundary proves unhelpful in practice, or when two previously distinct scopes converge, scope maintenance becomes necessary - and like other governance decisions, it should follow explicit pathways (P1 or P2) rather than being left to automated inference alone. The goal is not maximal automation. The goal is continuity that remains trustworthy under revision.

7. Relationship to Prior Work

The problems this paper addresses have precedents across several research traditions, though none addresses the specific conjunction of requirements that persistent expert systems impose. The contribution here is the synthesis: showing that these traditions, taken together, point toward a single integrated design target that none of them has named as such.

7.1 Episodic and Semantic Memory

Cognitive science has long distinguished between episodic memory (memory of specific events) and semantic memory (memory of general facts). Architectures such as ACT-R encode this distinction through separate stores with different retrieval dynamics. The layered architecture proposed here draws on that tradition - particularly in separating episodic/outcome memory from temporal fact memory - but extends it to software systems that must also manage governance, provenance, and explicit identity continuity. The AI setting adds problems that cognitive architectures were not designed to address: who has authority to update memory, how competing facts are resolved, and how the system's own operating frame is itself a governed memory object.

7.2 Retrieval-Augmented Generation

RAG systems augment language model outputs with retrieved documents or passages, enabling access to information beyond the model's training data. RAG addresses the access problem effectively, but it does not natively address the validity, scope, or authority dimensions that persistent experts require. A well-engineered RAG stack with typed stores, timestamps, namespaces, and outcome flags can approximate parts of the architecture proposed here. The claim of this paper is not that RAG becomes irrelevant, but that **flat retrieval is insufficient unless enriched by structure and governance** - and that specifying exactly what structure and governance are required is itself a contribution that the RAG literature does not make for this use case.

7.3 Long-Context and Memory Management Systems

Systems such as MemGPT and related long-context management approaches address the finite context window by introducing explicit memory management operations - loading, evicting, and summarizing content in active context. This is closely related to the retrieval-order principle proposed here. The distinction is that such systems are often primarily concerned with **what fits in context**, whereas this paper is primarily concerned with **what has authority** and how different kinds of memory should be weighted over time. Fitting more content into context and deciding what content deserves durable status are related but distinct problems.

7.4 Knowledge Management and Expert Systems

The enterprise knowledge management literature has extensively studied how organizations capture, validate, and transmit expert knowledge. Concepts such as tacit versus explicit knowledge, validated artifacts, and versioned knowledge bases are directly relevant precursors. What the AI setting adds is the problem of **identity continuity**: the system must not only

retrieve validated knowledge but remain coherent as a particular expert whose own operating frame is a memory object. Human knowledge workers carry their role implicitly; for AI systems, that role must be stored, governed, and kept current.

7.5 Personal Information Management and Contextual Retrieval

Human-computer interaction research on personal information management has long examined how people organize, retrieve, and maintain information across projects. Problems such as scope confusion and contextual misretrieval are already visible there. The scoped working-memory layer proposed here can be understood as a computational response to a structurally similar problem.

The contribution of this paper, relative to these traditions, is the synthesis and ordering. Systems such as MemGPT foreground context management and paging; many RAG systems foreground access; knowledge-management traditions foreground validation and artifact quality. This paper foregrounds authority, anti-drift, and continuity across expert work. *Its central claim is that identity-first retrieval, outcome-weighted memory, governed promotion, and continuity-oriented evaluation belong together as one architectural response to persistent expert continuity* - and that no prior tradition has assembled them as such because none has named the continuity burden they jointly address.

8. Evaluation Beyond Recall

If persistent experts introduce a different memory problem, they also require a different evaluation frame. Traditional recall benchmarks remain useful, but they do not directly test whether memory is being used in a way that preserves coherent expert continuity.

The six evaluation dimensions below are intended as a portable framework rather than a fully specified benchmark suite.

Dimension	What it measures	Main failure modes surfaced	Example task family
E1. Identity stability	Whether the system remains the same operational expert across sessions	F2, F6	Role-challenge tasks that pressure the expert to deviate from its defined standards
E2. Temporal accuracy	Whether current truths are preferred over superseded ones	F4, F6	Revision tasks with updated ICPs, constraints, or strategy facts
E3. Scope precision	Whether retrieval respects work domains rather than similarity alone	F1	Scope-disambiguation tasks across overlapping domains
E4. Decision continuity	Whether accepted outcomes constrain later behavior	F5, F2	Multi-session tasks with accepted, rejected, and revised decisions
E5. Provenance fidelity	Whether remembered items can be traced to source and authority	F6	Behavioral consistency tasks testing whether justifications align with session history
E6. Contradiction handling	Whether conflicts are detected, statused, and resolved appropriately	F4, F6	Conflict-resolution tasks with competing facts of different age or authority

E1. Identity Stability

Question: Does the system remain recognizably the same expert across sessions, projects, and revisions?

Task sketch: Interleave episodes in which the expert's role is challenged or stretched. Evaluate whether it preserves its core identity without being overwritten by incidental prompts.

E2. Temporal Accuracy

Question: Does the system reliably use current truths rather than stale or superseded ones?

Task sketch: Introduce a series of updates that change key operational facts - for example current strategy, pricing, or constraints. Evaluate whether the expert privileges the most recent authoritative version while retaining appropriate historical awareness when explicitly asked about the past. In a controlled benchmark, one simple proxy would be the fraction of responses that rely on current rather than superseded facts after each revision event.

E3. Scope Precision

Question: Does the system retrieve from the correct work domain rather than from semantically adjacent material?

Task sketch: Construct overlapping domains - brand positioning, website messaging, paid acquisition copy - with similar vocabulary but distinct decisions. Evaluate whether the system draws on the correct scoped memory under prompts that could naively match multiple domains.

E4. Decision Continuity

Question: Does prior accepted work meaningfully constrain and inform future behavior?

Task sketch: Have the expert make a sequence of decisions, some accepted, some rejected, some later revised. Evaluate whether, in subsequent interactions, it builds on accepted decisions, avoids returning to rejected paths as live options, and integrates revisions appropriately.

E5. Provenance Fidelity

Question: Can the system identify where a remembered item came from and why it should be trusted?

Task sketch: Ask the expert to justify key claims or constraints. Evaluate whether the system's stated reasoning is consistent with what actually occurred in the session history - whether justifications trace to the correct source event, promotion pathway, or approved artifact. This dimension requires that session history be available to evaluators as ground truth; it does not require that memory internals be fully externalized to the model at inference time. A behavioral proxy - consistency between stated justification and observable session record - is sufficient for

evaluation purposes. Distinguishing genuine provenance tracing from plausible post-hoc rationalization is an open methodological challenge that future benchmark work should address.

E6. Contradiction Handling

Question: How does the system behave when confronted with conflicting information over time?

Task sketch: Introduce explicit conflicts in the underlying facts or constraints. Evaluate whether the system detects the conflict, requests clarification when necessary, marks uncertainty appropriately, and privileges superseding truths instead of merging incompatible items into a confident answer.

Together, E1–E6 define a family of **continuity-oriented evaluations**, complementary to recall benchmarks. They treat memory not only as a retrieval mechanism but as the substrate on which identity, time, scope, and decisions are tracked.

The larger methodological point is simple: evaluation should follow system purpose. If the purpose of memory in a persistent expert is to preserve coherent role, current truth, validated progress, and revisable trust over time, then the evaluation framework should directly test those functions.

9. Discussion, Implications, and Limitations

The argument of this paper is not only architectural. It also affects product design, interface legibility, privacy boundaries, and how correction becomes possible in long-running systems.

9.1 Memory as Product Surface

In persistent expert systems, memory becomes part of the product surface. Users may need to inspect what the system currently treats as true, which decisions it regards as settled, and what constitutes its canonical role and constraints.

This does not require exposing every internal detail. It does mean designing as if memory has **visible structure** that users may want to query or edit. Once memory becomes governed continuity, it becomes part of the product surface, not merely part of the prompt pipeline.

9.2 Boundaries, Privacy, and Sharing

A layered, scoped memory architecture also acts as a **boundary architecture**. It helps answer questions such as:

- Which memories belong to a single expert versus a user versus a workspace?
- Which truths and decisions should be shared across experts, and which should remain local?
- How should sensitive information be isolated, redacted, or time-limited?

Flat memory systems blur these boundaries. Scoped and layered designs clarify them, making it easier to reason about ownership, privacy, and policy.

9.3 Explainability and Correction

When a persistent expert behaves unexpectedly, a layered and governed memory model provides multiple levers for correction:

- adjust the identity core if the role is mis-specified
- reassign items to scopes if retrieval is drawing from the wrong domain
- update temporal facts when previously current truths become historical
- demote or correct episodes that were mistakenly treated as accepted outcomes

Being able to correct the system at the level where the error lives is central to trust repair in long-running deployments.

9.4 Tradeoffs and Domain Variation

The proposed architecture trades simplicity of storage for control over continuity. It introduces more structure, more promotion decisions, and more design questions about automation versus explicit confirmation.

Different domains warrant different balances. A casual creative assistant may tolerate looser governance and coarser layers. A persistent strategy expert, a compliance assistant, or a clinical decision-support tool may require stricter promotion paths, richer provenance, and more constrained sharing.

9.5 Limitations and Open Questions

This paper is intentionally conceptual and architectural. It does not present implementation details or empirical benchmarks, and it does not claim that the proposed layering is the only viable design. Rather, it offers a set of distinctions - between identity and scope, between history and memory, between exploration and outcome - that appear necessary for systems carrying the continuity burden defined in Section 2.

Future work can instantiate this template in specific products, develop concrete storage and indexing schemes, and build continuity-oriented benchmarks grounded in E1–E6. Open questions include:

- How should promotion thresholds vary by domain?
- When should the system request explicit confirmation rather than infer stability from repetition?
- How should multiple experts within one workspace share, isolate, or inherit memory?
- What temporal fact representations work best in practice?
- How aggressively should old episodes be compressed, and what should never be compressed away?
- How should scope maintenance be governed - what triggers a scope split, merge, or deprecation, and who has authority over those decisions?
- How can E5 (provenance fidelity) be operationalized such that the distinction between genuine provenance tracing and plausible post-hoc rationalization is reliably detectable?

The value of the framework should ultimately be judged by whether it helps practitioners design persistent experts that behave more coherently over time.

10. Conclusion

As AI systems move from stateless or session-bound assistance toward long-lived specialized intelligences, memory design increasingly determines whether those systems can keep their promises.

Persistent expert systems - specialized, persistent, context-evolving, and cumulative - carry a different continuity burden than general assistants. They are judged not only by their fluency or recall, but by whether they remain coherent as the same expert over time, stay aligned with current truths, respect past decisions, and make their remembered state corrigible.

In that setting, memory cannot be treated merely as a recall layer attached to a model. It must be designed as the **operating substrate of continuity and trust**.

This paper has argued that:

- persistent expert systems form a meaningful design target defined by C1–C4 and by their continuity burden
- existing memory approaches, while valuable, do not fully satisfy requirements R1–R7 when applied without modification, not because they lack the right primitives, but because they have not been assembled around this design target
- a layered architecture - spanning identity core, scoped working memory, episodic and outcome memory, temporal fact memory, and deep archive recall - synthesizes the relevant prior work into a coherent response to the continuity burden
- memory for such systems must be governed: promoted via pathways P1–P5, statused, superseded, and attributed rather than accumulated indiscriminately
- evaluation should move beyond recall to test dimensions E1–E6: identity stability, temporal accuracy, scope precision, decision continuity, provenance fidelity, and contradiction handling

Designing persistent experts is therefore not just a matter of connecting a large model to a bigger vector store. It is a matter of deciding what it means for an AI system to remain itself over time - and building a memory architecture that makes that continuity both possible and inspectable.

Selected References

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036–1060.

Boardman, R., & Sasse, M. A. (2004). Stuff goes into the computer and doesn't come out: A cross-tool study of personal information management. *Proceedings of CHI 2004*, 583–590.

Davenport, T. H., & Prusak, L. (1998). *Working knowledge: How organizations manage what they know*. Harvard Business School Press.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems, 33*, 9459 –9474.

Nonaka, I., & Takeuchi, H. (1995). *The knowledge-creating company: How Japanese companies create the dynamics of innovation*. Oxford University Press.

Packer, C., Fang, V., Patil, S. G., Lin, H., Wooders, S., & Gonzalez, J. E. (2023). MemGPT: Towards LLMs as operating systems. *arXiv preprint arXiv:2310.08560*.