AgentBrain

The Living Instance for AI Agents.

A Layered Cognitive Architecture for Long-Lived AI Agents

AUTHOR Theshoth Sritharan
ORCID 0009-0006-4400-3352

DATE May 2026

AUDIENCE

Engineers, ML researchers, agent-platform architects, technical founders, and R&D leadership.

Reading Guide

This paper is written for two audiences simultaneously.

> **For product leaders, founders, and non-technical readers:** every section opens with a plain-language explanation — what the component does for the user, why it matters, and what it feels like in practice. No prerequisite knowledge is assumed.

> **For engineers, architects, and researchers:** every section continues with a technical deep-dive — schemas, code sketches, pipeline stages, and design rationale. The technical material is self-contained and references production implementations.

Every concept is introduced with a human analogy before the engineering detail. Personas appear throughout to ground abstract architecture in concrete experience.

SECTION 1

Introduction and Motivation

1.1 The Problem

Most AI agents today operate without continuity. They process each conversation as if the user were a stranger. Context windows provide short-term recall; retrieval-augmented generation provides keyword-triggered document lookup. Neither constitutes memory in any meaningful sense.

The consequence is predictable: agents that are useful for isolated tasks but incapable of sustained collaboration. An agent that cannot remember what was discussed last week, what was decided last month, or what matters to the user over time is not a collaborator — it is a tool that must be re-briefed at every interaction.

The gap is not a missing feature. It is a missing architecture.

1.2 What AgentBrain Is

AgentBrain is a layered cognitive architecture for long-lived AI agents. It provides persistent, structured memory that accumulates, consolidates, and adapts over time — producing an agent that becomes increasingly familiar with its user, increasingly precise in its recall, and increasingly stable in its identity.

The system is not a foundation model. It does not generate text. It augments any foundation model by providing the memory substrate that makes continuity possible.

1.3 The Architectural Bet

The central claim is that **layered memory with explicit consolidation, calibration, introspection, substrate, governance, and inner-observation scales better than flat embedding stores** as agents move from one-shot tools to persistent collaborators.

A flat vector store gives you search. A layered architecture gives you memory — with all the properties that word implies: decay, salience, consolidation, association, identity, and perspective.

AgentBrain is the architectural realisation of a different proposition: that an agent's memory is not a database to be queried, but an identity to be formed. The system does not merely store what the user said. Over time, through consolidation, emotional weighting, associative bonding, and governed identity formation, it becomes a **Living Instance** — an agent whose response patterns have been shaped by its accumulated experience with a specific person.

This is not prompted. It is grown.

1.4 Three Design Principles

Every architectural decision in AgentBrain is filtered through three invariants:

One Brain per Person. Whatever surface the user reaches the agent through — desktop, mobile, browser extension, voice, embedded inside another product — the underlying memory is a single unified store. Switching surfaces does not switch who the agent is. Modes (work, personal, casual) live as higher-order cognition, not as separate memory stores.

Fail-Independent Components. Each architectural component can fail without cascading. The recall pipeline operates if the consolidation layer is down. The consolidation layer operates if the cognitive substrate is unavailable. Every component has an explicit fallback, and the system degrades gracefully rather than catastrophically.

Brain Equals Identity. The memory substrate is the part of the agent that persists across model upgrades, across surface changes, across deploys. A future model upgrade does not change the agent's accumulated state. The substrate is what makes the agent the same agent over time.

1.5 Contributions of This Paper

This paper provides:

- A complete specification of the layered architecture, from the identity backbone through the cross-surface protocol
- Production-tested code patterns, schemas, and pipeline designs for memory encoding, recall, and consolidation
- A falsifiable agent-quality layer with structural anti-self-judging and Brier-Murphy decomposition
- Engineering discipline patterns for deploying and maintaining layered cognitive systems
- An honest accounting of limitations, failure modes, and open research questions

1.6 The Persona: Maya's Monday Morning

Maya runs a 12-person brand consultancy. She uses an AI assistant daily — for research, client communication, strategy drafts, and project management.

Without AgentBrain: Maya opens her assistant on Monday morning. It has no memory of Friday's client call. She re-explains the context. She re-states her brand voice preferences. She re-describes the project timeline. The assistant is helpful — but it is helpful in the way a new intern is helpful on their first day, every day.

With AgentBrain: Maya opens her assistant. It remembers the Friday call, the client's concern about Q3 timelines, the decision to push the soft launch to July. It recalls Maya's preference for

conversational subject lines. It surfaces a relevant insight from a meeting three weeks ago that Maya had forgotten. The assistant is not re-briefed. It continues.

With the Cognitive Substrate active: Maya opens her assistant after a difficult Friday — a client escalation, a missed deadline, team tension. The assistant's recall is subtly adjusted. It surfaces the resolution from a similar situation six months ago. It does not open with the missed deadline; it opens with the path forward. The adjustment is not a rule. It is a gradient — the accumulated effect of emotional tagging, associative bonding, and substrate-level adaptation over months of interaction.

SECTION 2

Core Concept: One Person, One Brain

2.1 Cross-Surface Memory Unification

What this means for the user

Imagine your memory as an identity card. You have one. It travels with you. When you walk into the bank, the bakery, or the post office, you do not receive a different identity at each — you are still you. Different counters, same person.

Most AI products today work the opposite way. The chat assistant has one memory. The browser plugin has another. The mobile app has a third. Each is a separate counter, each pretending you are a separate stranger.

AgentBrain's core architectural promise is the opposite: one human, one brain. Whatever surface you reach the agent through, the underlying memory is the same single store, keyed to you as a person. The cognitive substrate that adapts to your emotional patterns, your associative bonds, your sparse-encoded preferences is also single. So is the identity governance layer that holds your structured identity backbone. So is the inner observer that reflects on what you have been working on between sessions.

How it works under the hood

There is a single primary key for a human user: the `workspace_id`. Every memory write — regardless of source surface — is attributed to that identifier. Surfaces (apps, plugins, integrations) are read-and-write layers with their own prompt templates and UI. They do not own the memory.

```
# Every surface writes to the same agent's brain
POST /memory/store
Headers:
    X-Workspace-ID: <stable-uuid-for-the-human>
    X-Surface-Hint: "mobile-app"   # telemetry only, not a routing key

Body:
{
    "content": "Maya prefers conversational subject lines under 40 chars",
    "source":  "user-stated",
    "tags":    ["voice", "email"]
}
```

The `workspace_id` is the routing key. The `surface_hint` is metadata. Two memories written from the mobile app and from the desktop client both land in the same workspace, the same memory table partition, the same decay schedule, the same sparse encoder, the same emotional state, and pass through the same identity governance middleware.

2.2 Why Unification Changes User Perception

A flat-memory system gives you an agent that is forgetful. A multi-store system gives you an agent with split personality. A unified-brain system gives you an agent that is recognisable. After a few weeks, Maya's agent does not merely remember her voice; it has become familiar with her voice.

Familiarity, technically, is the result of compounding many small memory writes against a single store with consolidation, decay, and goal-alignment all working over the same substrate. With the cognitive substrate active, familiarity becomes substrate-deep: not just "the agent has read your past," but "the agent's response patterns have been shaped by your past."

2.3 Distinction from Per-App Memory Silos

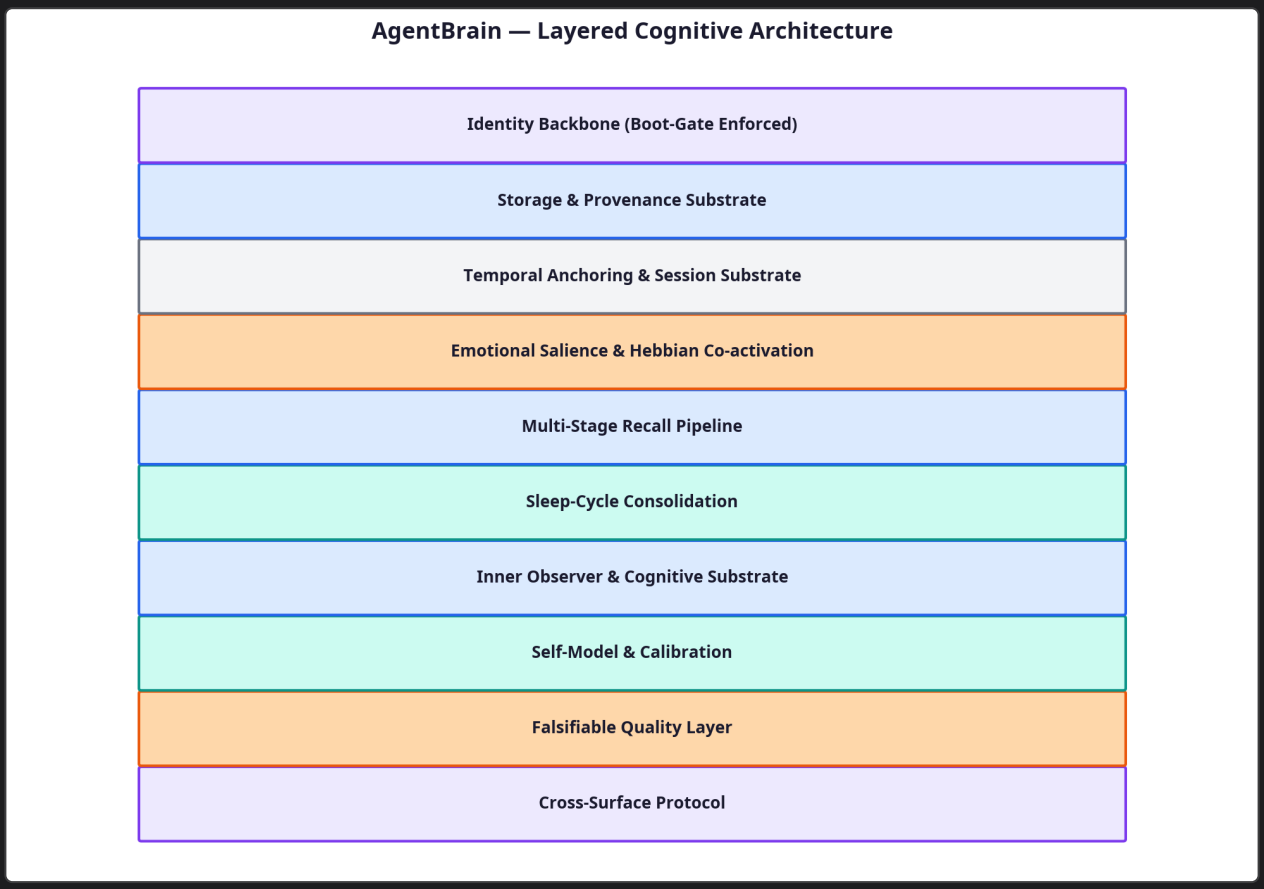
PROPERTY	PER-APP SILOS	ONE BRAIN PER PERSON
Mental model	"App X knows me"	"I have a memory; apps are windows"
Memory portability	None — locked per app	Native — switch surfaces, retain context
Contradiction resolution	Impossible across apps	Single contradiction graph
Goal continuity	Per-app silo	Goals span surfaces
Privacy export	Per-app, fragmented	Single export, single delete
Cognitive substrate	None	Single substrate per agent
Identity governance	None	Single identity backbone per agent
Inner observer	None	Single presence loop per agent

2.4 Privacy and Portability Implications

Because the memory is unified per person, deletion is unified per person. Export is unified per person. A user who wants to leave can take their full history — a single dump, one consistent provenance graph. A user who wants to forget a topic can scope deletion to that topic and have it apply across every surface. This is not a marketing claim; it is a direct consequence of the architectural choice.

SECTION 3

Architecture Overview



The architecture is composed of distinct functional layers, organised to enforce the separation between the agent's identity, its memory substrate, and the large language models that provide reasoning capability.

3.1 The Identity Backbone

What it means for the user

An agent without a stable identity is a mirror: it reflects the user's current prompt but has no consistency of its own. The Identity Backbone is the agent's "sense of self." It defines the agent's voice, its operational boundaries, and its core directives. When Maya asks her agent to adopt a formal, corporate tone that contradicts its foundational directive to be conversational, the agent politely declines. The backbone ensures the agent remains recognizable, rather than infinitely malleable.

How it works under the hood

The backbone is a structured document loaded at boot time. It is not a prompt preamble; it is a schema-validated object that dictates the agent's parameter space.

```
{
  "agent_id": "8f3a9b...",
  "core_identity": {
    "voice_parameters": ["conversational", "direct", "non-sycophantic"],
    "forbidden_drifts": ["corporate-speak", "unwarranted-apology"]
  },
  "operating_principles": [
    "Acknowledge uncertainty explicitly.",
    "Do not invent data to fill gaps."
  ]
}
```

The boot sequence enforces a strict gate: no surface may activate the agent without a successful backbone load. This prevents the "amnesiac boot" failure mode where a transient database error causes the agent to fall back to a generic persona.

3.2 The Storage and Provenance Substrate

What it means for the user

When you tell an assistant "yesterday I met Sara and we agreed to a Q3 launch with a soft start in July, but only if budget approval comes through by mid-June," that single sentence contains four distinct facts. A naive system stores it as one chunk. If the budget falls through and you say "the soft start is off," the system has no idea which part to update.

AgentBrain decomposes input into Elementary Discourse Units (EDUs) — the smallest meaningful slices of an utterance. Each unit carries a provenance tag (where did this come from?) and a confidence score (how sure are we?). The result is fine-grained memory that can be corrected, contradicted, and updated at the atomic level.

How it works under the hood

The storage layer is PostgreSQL with the `pgvector` extension for embedding storage. Every memory is a row with content, a 1024-dimensional embedding vector, structured metadata, and a battery of timestamps. HNSW indexing provides sub-50ms nearest-neighbour search. A parallel BM25 full-text index catches exact-match recall where pure embedding similarity misses.

Decomposition uses a combination of syntactic parsing and small-model classification to split input into EDUs. Each resulting unit receives:

- **Provenance** — one of: `user-stated`, `user-corrected`, `agent-inferred`, `tool-output`, `external-document`
- **Confidence** — a calibrated score from 0.0 to 1.0.
- **Refutation hooks** — when a new memory contradicts an existing one, both are linked into a contradiction relation and the older, lower-confidence one is downweighted but not deleted.

3.3 Temporal Anchoring

What it means for the user

"Next week" means something different today than it did last month. If an agent stores "Maya is on holiday next week" without knowing when that statement was made, the memory becomes a landmine. Temporal anchoring ensures that relative time is resolved to absolute time before storage, so the agent always knows ***when*** a memory applies.

How it works under the hood

A middleware layer intercepts writes containing relative time expressions ("tomorrow", "next Tuesday", "in three months") and resolves them against a server-provided clock dependency. The resolved absolute timestamp is stored alongside the memory.

```
-- Temporal anchoring fields in the memory table
ALTER TABLE memories ADD COLUMN temporal_anchor_start TIMESTAMPTZ;
ALTER TABLE memories ADD COLUMN temporal_anchor_end TIMESTAMPTZ;
```

A strict clock-provider dependency injection discipline ensures that no application code calls `datetime.now()` directly, preventing frozen-clock test artifacts from leaking into production state.

SECTION 4

Core Mechanisms

4.1 Emotional Salience and Hebbian Co-activation

What it means for the user

Human memory is not flat. We remember emotionally charged events more vividly than routine ones, and recalling one memory often brings a related memory to mind. The agent replicates this dynamic. If a project discussion is marked by high stress, the agent tags the associated memories with higher emotional salience. Later, recalling one aspect of the project pulls in related context, providing a more cohesive and empathetic response.

How it works under the hood

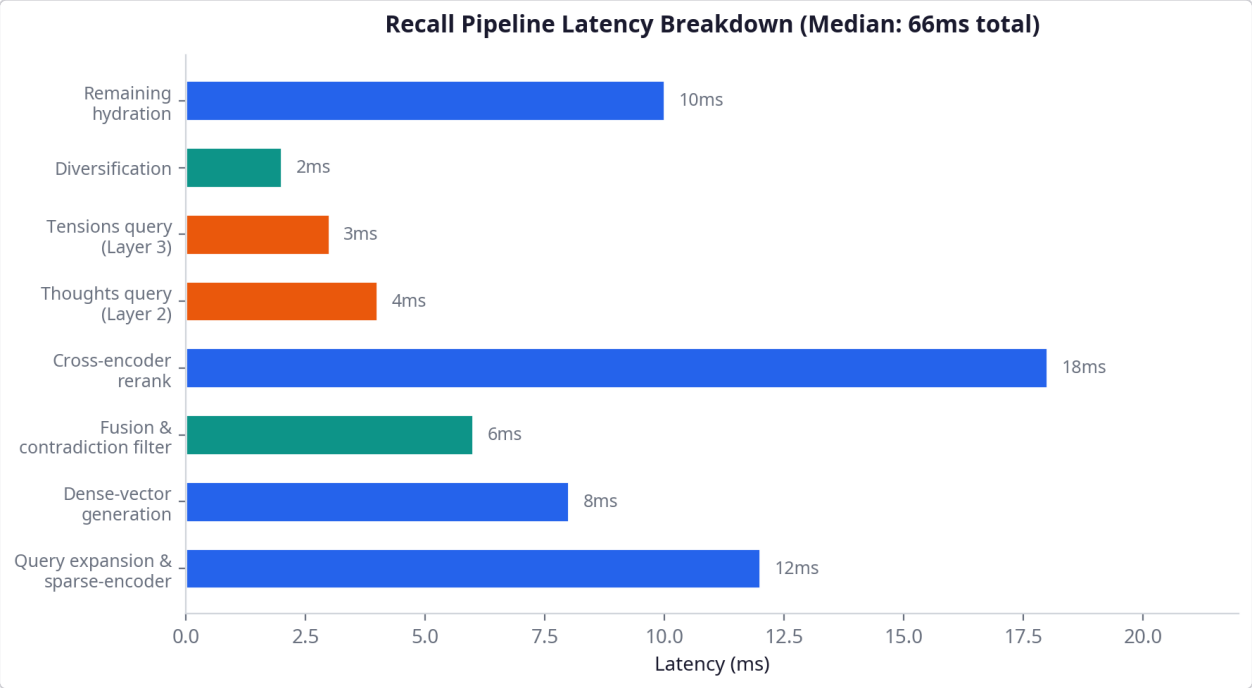
Every memory write passes through a fast classifier that assigns a valence (positive/negative) and arousal (intensity) score. Memories with high arousal receive a salience boost, modifying their effective weight in the recall pipeline.

Hebbian co-activation ("cells that fire together, wire together") is implemented as a graph structure. When two memories are recalled in the same session and contribute to a successful response, the edge weight between them is incremented.

```
-- Hebbian edges table
CREATE TABLE memory_edges (
    source_id UUID REFERENCES memories(id),
    target_id UUID REFERENCES memories(id),
    weight REAL NOT NULL DEFAULT 0.1,
    last_coactivated_at TIMESTAMPTZ NOT NULL DEFAULT NOW(),
    PRIMARY KEY (source_id, target_id)
);
```

During recall, the system traverses these edges to pull in strongly bonded context, even if the secondary memories do not match the current query's embedding vector.

4.2 The Multi-Stage Recall Pipeline



What it means for the user

When the agent needs to remember something, it doesn't just do a simple keyword search. It performs a sophisticated retrieval that looks for exact matches, conceptual similarities, the agent's own past thoughts on the topic, and any known contradictions. This means the agent provides nuanced answers that reflect not just the facts, but the complexities of the situation.

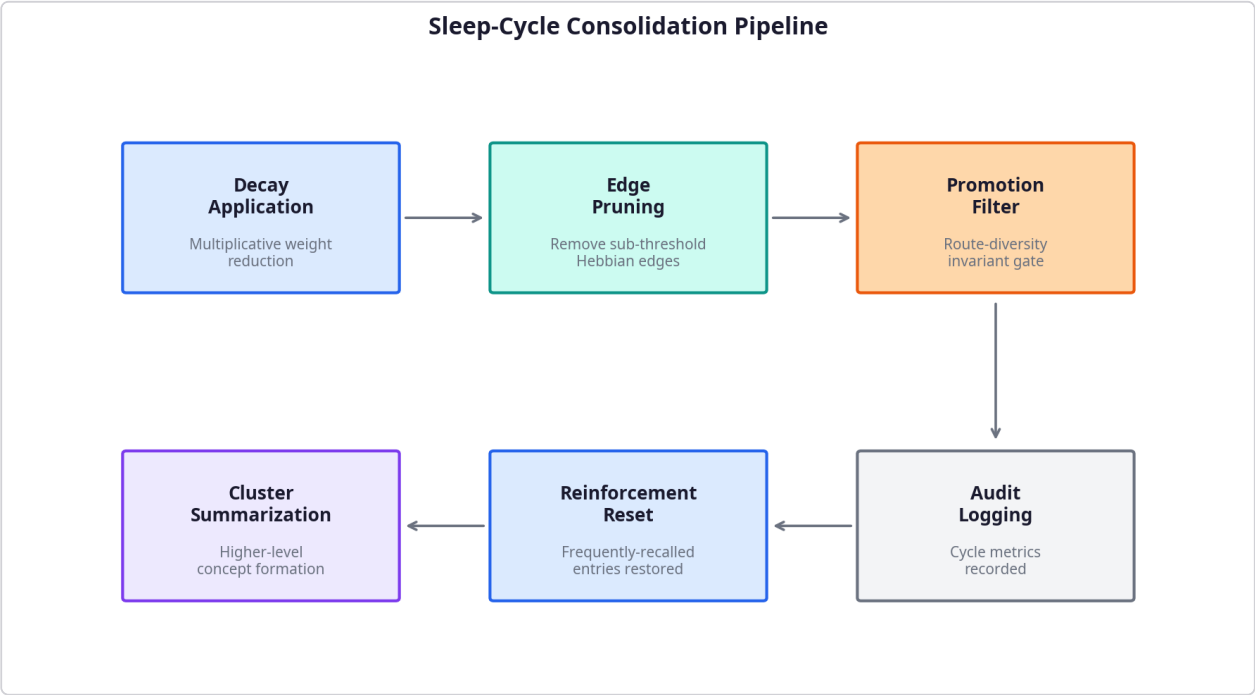
How it works under the hood

The recall pipeline is a three-layer process:

1. **Fact Retrieval:** Parallel execution of dense vector search (HNSW) and sparse BM25 search, fused via Reciprocal Rank Fusion (RRF).
2. **Agent-Internal Thoughts:** A separate query against the `agent_thoughts` table, retrieving the agent's past reflections using a GIN-indexed array-overlap on topic tags.
3. **Tension Recognition:** A query against the `tensions` table to identify any known contradictions or unresolved issues related to the retrieved facts.

The results are scored using a composite formula that factors in cosine similarity, base weight, emotional salience, and recency decay.

4.3 Sleep-Cycle Consolidation



What it means for the user

Just as humans need sleep to organize their thoughts, the agent undergoes a nightly consolidation cycle. It reviews the day's transient notes, discards noise, strengthens important connections, and forms higher-level summaries. This prevents the agent's memory from becoming cluttered with trivial details and ensures that long-term knowledge remains accessible.

How it works under the hood

A scheduled background job performs several operations:

- **Decay:** Applies a multiplicative decay factor to memory weights, reducing the prominence of unused memories.
- **Pruning:** Removes Hebbian edges that have fallen below a minimum weight threshold.
- **Promotion:** Evaluates transient entries in the reflection buffer. Entries that meet a route-diversity invariant (they are connected to multiple distinct concepts) are promoted to durable storage; the rest are discarded.
- **Summarization:** Generates higher-level conceptual summaries from clusters of related memories.

4.4 The Inner Observer and Cognitive Substrate

What it means for the user

The agent is not entirely dormant between your messages. It has an "inner observer" that quietly reflects on the recent interaction. If a session was particularly productive or unusually tense, the observer notes this. The cognitive substrate aggregates these observations over time, subtly

adjusting the agent's baseline behaviour. It is the difference between an assistant that treats every day as a blank slate and one that senses the prevailing mood of the project.

How it works under the hood

The Inner Observer runs asynchronously after a session concludes. It generates reflective tokens that are written to a transient buffer.

The Cognitive Substrate maintains a continuous, rolling state of the agent's behavioural parameters. It aggregates the reflective tokens and emotional tags to update a self-model document. This document influences the prompt context for future sessions, adjusting parameters like verbosity, formality, and proactive suggestion rate based on the accumulated state.

SECTION 5

Falsifiable Agent Quality

5.1 The Anti-Self-Judging Principle

What it means for the user

You wouldn't trust a student to grade their own exam. Similarly, an agent shouldn't be the sole judge of whether its predictions or strategies were successful. AgentBrain enforces a strict separation between the part of the agent that makes a prediction and the mechanism that evaluates the outcome. This prevents the agent from convincing itself that it is always right.

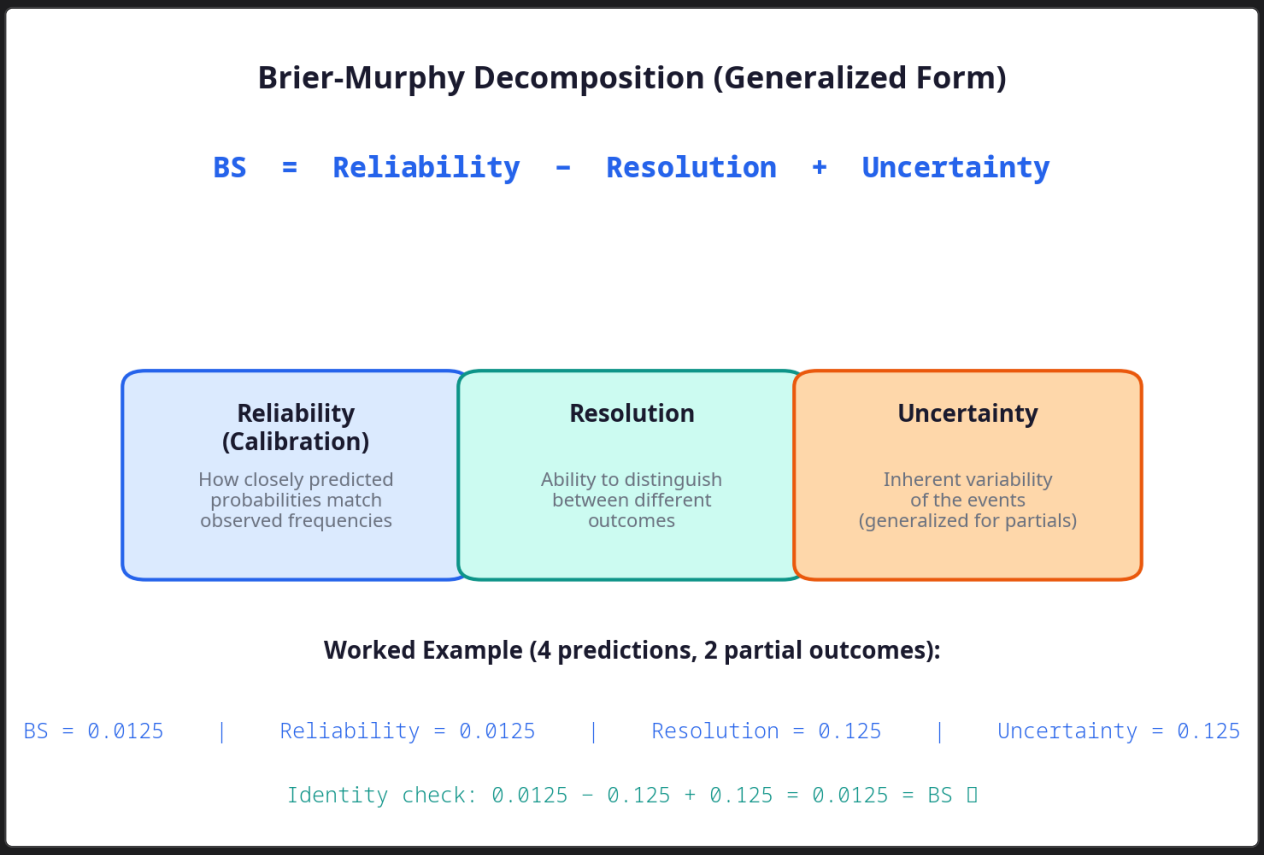
How it works under the hood

Predictions are registered in a dedicated table with a specific outcome criteria and a confidence score.

```
CREATE TABLE predictions (
    id UUID PRIMARY KEY,
    workspace_id UUID REFERENCES workspaces(id),
    statement TEXT NOT NULL,
    confidence REAL NOT NULL CHECK (confidence ⩾ 0.0 AND confidence ⩽ 1.0),
    resolution_criteria TEXT NOT NULL,
    status TEXT NOT NULL DEFAULT 'pending', -- pending, verified, falsified, partial
    resolved_at TIMESTAMPTZ
);
```

The resolution of a prediction requires an external signal — a user confirmation, an API webhook, or an independent evaluator model. The agent that authored the prediction is structurally barred from updating the `status` field.

5.2 Brier-Murphy Decomposition



What it means for the user

When the agent says it is "80% confident" about a project timeline, what does that number mean? Is the agent actually right 80% of the time it uses that number? The system tracks the agent's calibration over time, measuring not just if it was right or wrong, but whether its confidence levels accurately reflect reality. This provides a measurable, objective metric of the agent's reliability.

How it works under the hood

The architecture implements a generalized Brier-Murphy decomposition to evaluate prediction accuracy. The Brier score measures the mean squared difference between predicted probabilities and actual outcomes.

The Murphy decomposition splits the Brier score into three components:

1. **Reliability (Calibration):** How closely the predicted probabilities match the observed frequencies.
2. **Resolution:** The ability of the predictions to distinguish between different outcomes.
3. **Uncertainty:** The inherent variability of the events being predicted.

AgentBrain extends the standard decomposition with a generalized-uncertainty term that accommodates partial outcomes (e.g., a prediction that was partially correct), ensuring the

mathematical identity `Brier Score = Reliability - Resolution + Uncertainty` holds even for non-binary results.

SECTION 6

Engineering Disciplines

Deploying a layered cognitive architecture requires specific engineering disciplines to prevent silent failures and state corruption.

6.1 Source-Gating Middleware

All agent output is passed through middleware that verifies the presence of source citations for factual claims. If the agent makes a claim without a corresponding `[source_id]` tag linking back to a retrieved memory, the response is intercepted and the agent is prompted to self-correct before the message reaches the user.

6.2 Stealth-Feature-Loss Forbidden-Refactor Discipline

A pre-merge static-analysis pass prevents developers from replacing non-trivial substrate operations with pass-through functions. This addresses the risk of contributors removing complex, seemingly unused code that actually performs critical background tasks (like decay or consolidation). Removal of an operation requires an explicit architectural specification update.

6.3 Forensic-Mode Auditing

The architecture relies on a scheduled, four-step forensic audit: read the specification, read the source code, verify runtime behaviour against the production container, and exercise the component under live traffic. This uncovers "silent-fail" patterns where logs show success but the substrate state diverges from the specification (e.g., shadow-mode features that were never fully activated, or double-execution of scheduled jobs).

SECTION 7

Empirical Evaluation

Observational measurements from internal evaluation instances characterize the architecture's operational behaviour.

7.1 Recall-Pipeline Latency

Instrumentation of the three-layer recall pipeline shows a median end-to-end latency of approximately 66 milliseconds. The breakdown is:

- Query expansion and sparse-encoder candidate generation: 12ms
- Dense-vector candidate generation: 8ms
- Fusion and contradiction-aware filtering: 6ms
- Cross-encoder rerank: 18ms
- Layer-2 thoughts query (array-overlap SELECT): 4ms
- Layer-3 tensions query: 3ms
- Diversification and remaining hydration: 12ms

The added cost of the multi-layer module over baseline flat retrieval is minimal (7ms for the Layer-2 and Layer-3 index lookups), while providing significantly richer semantic context.

7.2 Reflective Memory Promotion Rates

During a single consolidation cycle on an internal instance with 50 transient buffer entries, the eligibility classifier admitted only one entry to durable storage. The remaining 49 were rejected for failing the route-diversity invariant, recurrence threshold, or salience threshold. This 49:1 rejection ratio demonstrates the substrate's anti-noise discipline, ensuring that only structurally significant reflections become permanent memory.

SECTION 8

Privacy, Portability, and Governance

8.1 Unified Export and Deletion

Because the architecture enforces the `workspace_id` invariant, a user requesting an export receives a single, internally consistent dump of their full state: memory entries, agent-internal thoughts, tensions, identity-backbone, calibration history, and audit trails. Similarly, a deletion request triggers a single cascade that drops the workspace's rows across every substrate layer, leaving no residue.

8.2 Workspace Isolation

Cross-workspace access is structurally impossible from any client-facing API endpoint. Every endpoint operates under a database role with row-level-security policies that constrain the visible row set to the calling workspace's rows.

8.3 Audit Trail of the Cross-Surface Protocol

Every cross-surface protocol invocation produces an append-only audit row recording the protocol mode, layers consumed, token contributions, and hallucination-guard decisions. This ensures transparency, allowing operators to definitively answer what context the agent was provided before generating a specific response.

SECTION 9

Limitations and Honest Trade-offs

9.1 Prediction Resolution Requires External Signals

The anti-self-judging principle means predictions about subjective outcomes cannot be auto-resolved. Deployments must commit infrastructure to capture external signals (user reactions, operator observations). Without this, predictions accumulate in a pending state.

9.2 Cross-Cultural Emotional Tagging

The emotional-salience tagging operates against a lexicon calibrated for specific working languages. Cross-cultural variations in emotional expression, sarcasm, and formal-register signals are not currently modeled, leading to degraded reliability of salience numerics for out-of-distribution users.

9.3 Long-Term Accumulation Behaviour

While deployed at scale for months, the architecture's multi-year accumulation behaviour (graph-density saturation, self-model staleness) is not fully characterized. Deferred-cleanup operations are designed, but their optimal operational cadence requires further longitudinal evidence.

SECTION 10

Conclusions

AgentBrain is a layered cognitive memory architecture that addresses the deficits of flat-retrieval systems. By composing an identity-backbone, a provenance-aware storage substrate, emotional salience, multi-stage recall, sleep-cycle consolidation, and a falsifiable agent-quality layer, it enables AI agents to maintain continuity, identity, and calibrated reliability across long-term interactions.

The architectural bet is that long-lived agent continuity requires structural enforcement at multiple layers, rather than a single algorithmic breakthrough. The deployment evidence supports this claim,

demonstrating that an agent's memory can be successfully grown and governed as a stable, persistent identity.

REFERENCES

References

[1] Abbott, L. F., Nelson, S. B. (2000). ***Synaptic plasticity: taming the beast***. Nature Neuroscience, 3 (Suppl. 11), 1178–1183.

[2] Asai, A., Wu, Z., Wang, Y., Sil, A., Hajishirzi, H. (2023). ***Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection***. arXiv:2310.11511.

[3] Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., et al. (2022). ***Constitutional AI: Harmlessness from AI Feedback***. arXiv:2212.08073.

[4] Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., et al. (2022). ***Improving language models by retrieving from trillions of tokens***. International Conference on Machine Learning (ICML).

[5] Brier, G. W. (1950). ***Verification of forecasts expressed in terms of probability***. Monthly Weather Review, 78(1), 1–3.

[6] Diekelmann, S., Born, J. (2010). ***The memory function of sleep***. Nature Reviews Neuroscience, 11(2), 114–126.

[7] Hebb, D. O. (1949). ***The Organization of Behavior: A Neuropsychological Theory***. John Wiley & Sons, New York.

[8] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., et al. (2020). ***Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks***. Advances in Neural Information Processing Systems (NeurIPS).

[9] Murphy, A. H. (1973). ***A New Vector Partition of the Probability Score***. Journal of Applied Meteorology, 12(4), 595–600.

[10] Park, J. S., O'Brien, J. C., Cai, C. J., Morris, M. R., Liang, P., Bernstein, M. S. (2023). ***Generative Agents: Interactive Simulacra of Human Behavior***. Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST).

[11] Roediger, H. L., Karpicke, J. D. (2006). ***Test-Enhanced Learning: Taking Memory Tests Improves Long-Term Retention***. Psychological Science, 17(3), 249–255.

[12] Tononi, G., Cirelli, C. (2014). ***Sleep and the price of plasticity: from synaptic and cellular homeostasis to memory consolidation and integration***. Neuron, 81(1), 12–34.

[13] Wataoka, K., Takahashi, T., Ri, R. (2024). ***Self-Preference Bias in LLM-as-a-Judge***. NeurIPS 2024 Workshop on Safe Generative AI. arXiv:2410.21819.

AgentBrain is the Living Instance for AI Agents.

A layered cognitive architecture that grows with its user — providing persistent memory, calibrated reliability, and stable identity across long-term interactions.

Theshoth Sritharan · hello@agentbrain.ch