

Architecting Collective Intelligence: A Hybrid Memory Framework for Scalable Multi-Agent Collaboration

Architectural Foundations for Distributed Cognition

The design of a scalable memory system for Artificial General Intelligence (AGI) agents engaged in multi-agent collaboration necessitates an architectural foundation that prioritizes modularity, hierarchy, and robust inter-agent communication protocols. Moving beyond monolithic, agent-centric designs, the most effective frameworks are those engineered for distributed cognition, where memory is treated as a shared system capability rather than a confined individual attribute ³. This approach allows rationale, assumptions, and insights to flow and be reused across complex systems and organizational timescales, enhancing both individual adaptability and collective intelligence ³. The primary challenge in such environments is ensuring "multi-agent memory consistency," defined as the fundamental requirement for all agents to maintain a coherent and up-to-date view of the shared memory state, a problem identified as the most pressing open challenge in current multi-agent memory architectures ¹. To address this, two critical protocol gaps have been identified: cache sharing protocols for efficient data exchange between agents' caches, and structured memory access control mechanisms to manage permissions and interactions with shared memory spaces ¹.

A core principle for achieving scalability is the adoption of a hierarchical architecture. Drawing parallels from computer architecture, a three-layer memory hierarchy comprising Input/Output (I/O), cache, and main memory provides a powerful blueprint for managing the growing complexity of memory requirements in collaborative LLM systems ¹. This structure creates layers of abstraction that separate concerns of speed, volatility, and capacity. At the lowest level, fast but volatile caches can handle immediate, high-frequency interactions, while at the highest level, persistent databases serve as the main memory for long-term, shared state ². This layered separation directly addresses the need to manage memory requirements across multiple agents by providing a scalable foundation for data exchange and storage ¹. The MemOS framework further operationalizes this concept by unifying heterogeneous memory resources under a single system philosophy, demonstrating the power of a hierarchical approach in managing

diverse memory types ⁶ . Similarly, the Hierarchical Agent Orchestration pattern reinforces the value of layering by using a top-level 'planner agent' to decompose high-level objectives into subtasks executed by specialized 'worker agents' ² . This not only enhances modularity and simplifies debugging but also improves overall system transparency and scalability, principles that are directly transferable to the design of a memory architecture ² .

Modularity is another critical architectural tenet, enabling specialization and resilience. At the system level, modularity is exemplified by agent orchestration patterns like Sequential Agent Chains and Shared State via Persistent Databases ² . While sequential chains are optimal for deterministic processes, their fragility makes them unsuitable for the dynamic nature of collaborative AGI. In contrast, the shared state paradigm, which uses a centralized data store like Redis or a vector database, enables asynchronous interaction among agents ² . However, this approach introduces significant challenges related to race conditions and data inconsistency, underscoring the necessity of rigorous consistency protocols like locking mechanisms or optimistic concurrency control to ensure an accurate shared state ² . Beyond system-level organization, modularity must also be applied internally to the memory system itself. The Contextual Memory Intelligence (CMI) framework proposes an "Insight Layer" composed of five distinct, interoperable modules designed to work in concert: a Context Extractor, Insight Indexer, Drift Monitor, Regeneration Engine, and Reflection Interface ³ . This modular design allows for focused development, debugging, and optimization of each specific function, preventing the creation of a brittle, monolithic memory core ³ . Such a modular internal structure is essential for building a memory system that can evolve and scale without compromising its integrity.

The successful implementation of a distributed memory architecture hinges on well-defined protocols for inter-agent communication and data exchange. The MAICC framework, designed for cooperative Multi-Agent Reinforcement Learning (MARL), highlights the importance of retrieving relevant trajectories from memory as context, combining team-level task information with agents' current sub-trajectories during execution ⁵ . This implies a need for sophisticated query and retrieval mechanisms that can synthesize both global and local information. For broader collaboration, the MemOS framework introduces a MemStore, an open-access interface for controlled publishing, subscription, and distribution of memory units among agents ⁶ . This mechanism allows users to declare memory as publishable, define visibility and access rules, and enables consumers to subscribe to updates based on tags or semantic filters ⁶ . This formalizes the process of memory exchange, transforming it from an ad-hoc interaction into a managed, governed system capability. The effectiveness of these protocols is paramount;

without them, the risk of fragmented, inconsistent, and ultimately unreliable collective intelligence remains high. Therefore, any novel framework must treat the design of these inter-agent protocols as a first-order concern, integrating them deeply into the architectural DNA of the system. The combination of a hierarchical structure, internal modularity, and robust communication protocols forms the foundational bedrock upon which a truly scalable and collaborative AGI memory system can be built.

Architectural Pattern	Description	Key Advantage	Key Disadvantage
Hierarchical Architecture	Organizes memory into distinct layers (e.g., I/O, Cache, Main Memory) based on performance characteristics 1 6 .	Manages complexity and scales by separating concerns of speed, volatility, and capacity.	Can introduce latency between layers if not managed efficiently.
Hierarchical Agent Orchestration	A top-level 'planner agent' delegates tasks to specialized 'worker agents' 2 .	Enhances modularity, simplifies debugging, increases transparency, and improves scalability 2 .	Relies on the efficacy of the central planner; potential bottleneck.
Shared State via Persistent Database	Agents interact asynchronously through a centralized data store (e.g., Redis, vector DB) 2 .	Enables flexible, non-blocking interaction and persistence across agent lifecycles.	Requires rigorous consistency protocols (e.g., locking, optimistic concurrency) to prevent race conditions 2 .
Contextual Memory Intelligence (CMI) Layer	An internal modular architecture for memory-as-infrastructure, with dedicated modules for extraction, indexing, drift monitoring, etc. 3 .	Allows for focused development and optimization of specific memory functions, improving maintainability.	Adds complexity in inter-module communication and coordination.
MemStore (MemOS)	An open-access interface for controlled publishing, subscription, and distribution of memory units among agents 6 .	Formalizes and governs memory exchange, supporting collaboration across models and institutions.	Requires robust permissioning and governance mechanisms to be secure and effective.

A Hybrid Memory Model for Dynamic Knowledge Representation

To construct a memory system capable of supporting the multifaceted demands of AGI agents in collaborative settings, a hybrid memory model is indispensable. Relying on a single type of memory representation, whether purely textual or parametric, imposes significant limitations that hinder scalability, efficiency, and adaptability. The provided literature establishes a clear dichotomy between these two primary forms of memory, highlighting their respective strengths and weaknesses. Textual memory, which includes explicit knowledge modules represented as passages or structured data, offers superior interpretability, easier implementation, and faster read-write efficiency 7 . It is highly

suitable for storing recent interactions, procedural steps, and user-generated content because it is straightforward to create and modify ⁷. However, its primary drawback is its inefficiency during retrieval; incorporating large amounts of textual memory into an LLM's context window increases inference cost and latency due to the quadratic growth of attention computation with sequence length, and it risks truncation beyond the model's context limit ⁷.

In stark contrast, parametric memory encodes knowledge directly into the weights of the LLM itself ⁷. This approach is exceptionally efficient for reading, as retrieved information does not incur additional context cost during inference, and it supports higher information density ⁷. Methods for creating parametric memories include supervised fine-tuning (SFT) or parameter-efficient techniques like LoRA to infuse domain-specific knowledge, resulting in highly specialized agents ⁷. Another approach, memory editing, aims to change only specific facts within the model's parameters, leaving other knowledge intact, making it more suitable for small-scale, real-time modifications ⁷. The critical disadvantages of parametric memory lie in its high cost and difficulty of writing. Fine-tuning is computationally expensive, time-consuming, and prone to catastrophic forgetting, where the model loses previously learned information when trained on new data ⁷. Memory editing methods, while more targeted, often require costly meta-training phases and face challenges in preserving unrelated memories ⁷. This fundamental trade-off between easy writing and efficient reading dictates that no single form can suffice for a comprehensive AGI memory system.

The most advanced and promising approach is to integrate multiple memory types into a unified, fluid system. The MemOS framework provides a compelling model for this by defining three distinct yet interconnected memory types that represent a semantic evolution trajectory ⁶. These are Plaintext Memory (explicit, dynamically retrievable knowledge), Activation Memory (the intermediate states, primarily the Key-Value cache, used for efficient modeling of long-range dependencies), and Parametric Memory (knowledge encoded in model weights) ⁶. A novel framework must adopt and expand upon this hybrid model. By treating these memory forms as first-class citizens, the system can leverage the best properties of each. For example, newly acquired, detailed factual information can be stored as plaintext for easy human review and modification. Frequently accessed or temporally proximate information can be loaded into activation memory (the KV-cache) for rapid recall during ongoing reasoning processes. Finally, stable, high-value knowledge that has proven its worth over time can be distilled and integrated into the parametric base model to provide a permanent, low-latency enhancement to the agent's core capabilities ⁶.

This hybrid model extends beyond just these three types. The State-Aware Memory (SAMem) paradigm introduces the concept of organizing memory at the level of 'state-specific reasoning thoughts' rather than globally shared experiences ⁴. This suggests a fourth, more abstract layer of memory that captures the nuances of decision-making contexts. Furthermore, the use of external tools allows for the acquisition of up-to-date information from sources like APIs, which must then be transformed and integrated into the agent's memory ⁷. Models like Toolformer are designed to teach LLMs how to use these tools effectively ⁷. This requires the memory system to not only store the final result but also the process of how that information was acquired, validated, and structured. Therefore, a truly comprehensive hybrid model must be able to represent not just static facts, but also dynamic processes, contextual states, and the provenance of information. The MemCube, the foundational unit of MemOS, is designed to encapsulate both memory content (Payload) and rich metadata, including descriptive identifiers, governance attributes, and behavioral usage indicators, enabling this kind of rich, heterogeneous representation ⁶. By standardizing these heterogeneous resources into a common format, the framework can facilitate seamless transitions between memory types and support a wide range of cognitive tasks, from simple fact recall to complex, multi-step reasoning grounded in a rich, evolving knowledge base.

Learning and Evolutionary Mechanisms for Continuous Adaptation

For an AGI memory system to remain effective in dynamic, collaborative environments, it must possess robust learning and evolutionary mechanisms that enable continuous knowledge integration and adaptation. Passive archiving of information is insufficient; the memory must actively participate in the reasoning cycle, transforming raw data into structured, actionable insight. The literature points towards several key primitives for achieving this, including structured interaction capture, active tool use for memory transformation, and a dynamic, policy-driven lifecycle management system that governs how knowledge evolves over time. A foundational step is moving beyond naive approaches to memory storage. The "complete interaction memory" strategy, which involves storing all historical agent-environment interaction data verbatim, is computationally prohibitive for long-term tasks due to the quadratic growth of attention computation costs with sequence length and the risk of exceeding LLM context limits ⁷. While this method maintains comprehensive information, its high computational overhead makes it impractical for real-time applications ⁷.

More sophisticated learning mechanisms focus on what information to keep and how to structure it. The distinction between "recent interaction memory" and "retrieved interaction memory" is crucial ⁷. Recent interaction memory, based on the Principle of Locality, stores only the most recently acquired data, such as observations from recent timesteps ⁷. While this is computationally efficient, it carries the significant risk of losing potentially crucial information from distant past interactions, which could be vital for long-term strategic planning or understanding complex causal chains ⁷. Conversely, retrieved interaction memory attempts to select content based on its relevance, importance, or topic, rather than simply recency ⁷. This is typically implemented by generating embeddings for memory entries during the writing phase and using these vectors to calculate matching scores during retrieval ⁷. Techniques like dual-tower dense retrieval with FAISS indexing (as in MemoryBank) or locality-sensitive hashing (as in RET-LLM) enable this relevance-based selection, ensuring that key information from the distant past remains accessible ⁷. This underscores the necessity of a structured indexing system, such as the one proposed by the CMI framework's Insight Indexer, which links insights across systems using vector and graph-based models to preserve longitudinal coherence ³.

A critical component of continuous adaptation is the active role of external tools in shaping the memory. The concept of "Memory Transformation through Tool Use" elevates the memory system from a passive repository to an active participant in the agent's cognitive process ². Inspired by methodologies like Toolformer, agents can use external tools to perform operations that refine and structure their internal or shared memory ². For instance, an agent could use a summarization tool to convert a lengthy log of interactions into a concise, structured summary, or employ an analytical tool to generate new insights from a collection of unstructured data points ². This process transforms raw data into a more valuable knowledge asset. The effectiveness of this approach depends critically on the quality of the tools selected and the validation of their outputs for reliability ². Integrating this capability requires the memory system to not only store the final output of a tool but also to record the action taken, the input provided, and the confidence of the result, thereby preserving a traceable rationale for future reflection and audit. This aligns with the CMI paradigm, which emphasizes capturing and regenerating context across time and roles to enable explainable and responsible decision-making ³.

Ultimately, the most sophisticated learning mechanism is a dynamic, policy-driven lifecycle management system that governs the evolution of memory. The MemOS framework provides a concrete model for this through its Policy-Aware Scheduling ⁶. This system enables dynamic transitions between memory types based on usage patterns,

allowing memory to evolve toward its optimal invocation form 6 . For example, frequently used plaintext can be promoted to activation vectors for faster access or even distilled into the parametric form for permanent integration 6 . Conversely, outdated or infrequently accessed parametric knowledge can be demoted back to plaintext storage to conserve resources 6 . This creates a fluid, adaptive memory landscape that continuously optimizes itself for performance and relevance. The transitions between states are driven by policies and heuristics based on factors like recency, access frequency, and contextual salience 6 . This evolutionary perspective treats memory not as a fixed archive but as a living system that grows, adapts, and refines itself over time, a core tenet of a truly intelligent memory framework. By combining structured retrieval, active tool use, and a dynamic scheduling policy, the memory system becomes a powerful engine for continuous learning and adaptation, essential for AGI agents operating in complex, collaborative domains.

Lifecycle Management and Retention Strategies for Coherent Memory

In a system designed for continuous learning and adaptation, retaining all information indefinitely is not only impossible but also counterproductive. An effective memory system must implement intelligent lifecycle management and retention strategies that balance the permanence of critical knowledge with the decay of irrelevant or outdated information. This requires moving beyond simple storage to a proactive system of governance that manages the evolution of memories from their initial creation to their eventual archival or deletion. The MemOS framework offers a powerful and operationalizable model for this through its four-state finite state machine: Generated, Activated, Merged, and Archived 6 . This structure provides a clear, policy-driven pathway for every piece of memory, ensuring that resources are allocated efficiently and that the overall memory state remains coherent and useful. Transitions between these states are not arbitrary but are driven by heuristics based on objective measures such as recency, access frequency, and contextual salience, forming the basis of a rational retention strategy 6 .

The lifecycle begins with the **Generated** state, where newly created memory entries reside. At this stage, the information is fresh but may not yet have established its value or relevance. From here, it can transition to the **Activated** state if it is accessed frequently, indicating high utility for ongoing tasks 6 . This might involve loading the memory into a

fast-access cache or the model's KV-cache to minimize retrieval latency 6 . If a piece of knowledge proves to be stable and consistently valuable over a longer period, it can be moved to the **Merged** state, where it is integrated into the parametric base model 6 . This permanent integration ensures that the knowledge is available at zero inference cost and contributes to the agent's core capabilities 7 . Finally, memories that are determined to be low-value, redundant, or outdated can be demoted to the **Archived** state, where they are moved to cheaper, persistent storage or deleted entirely, freeing up resources for new learning 6 . This systematic management of memory states is essential for maintaining a scalable and performant system, directly addressing the resource allocation challenges inherent in AGI.

Beyond simple resource management, a sophisticated retention strategy must explicitly account for the degradation of meaning and context over time. The CMI framework introduces two powerful concepts for this purpose: "Insight Drift" and "Contextual Entropy" 3 . Insight Drift quantifies the cumulative semantic misalignment between an original rationale and its later interpretations, while Contextual Entropy measures the degradation of memory coherence in distributed systems 3 . These are not merely academic metrics; they point to a critical failure mode in collaborative AI systems. Without a mechanism to detect and correct for drift, flawed logic or outdated assumptions can propagate silently through a team of collaborating agents, leading to systemic errors and a breakdown in collective intelligence. To combat this, the proposed framework should incorporate a dedicated "Drift Monitor" module, as envisioned in the CMI paradigm 3 . This module would continuously track stored insights, using techniques like vector similarity and temporal thresholds to detect when an idea has become misaligned with its original intent or with current reality 3 . Upon detecting significant drift, the system could trigger a "Resonance Intelligence" response, flagging the issue for human-in-the-loop review or initiating a regeneration process to restore coherence 3 . This proactive monitoring and correction mechanism is vital for maintaining the long-term health and reliability of the shared memory.

Furthermore, the framework must support mechanisms for human oversight and intervention, a feature emphasized in both Human-in-the-Loop (HITL) architectures and the CMI Reflection Interface 2 3 . HITL is essential in sensitive or regulated environments, such as the automated generation of legal briefs or medical diagnostic reports, where expert human verification and approval are mandatory 2 . The memory system must be designed to efficiently manage asynchronous human inputs without disrupting workflow continuity, often requiring intelligent queuing mechanisms or notification systems to bring issues like detected drift or contradictory memories to the attention of a human supervisor 2 3 . The Reflection Interface proposed by CMI

provides a formal mechanism for this, enabling human feedback, coherence recovery, and updates to memory scoring and surfacing logic ³ . This creates a closed loop of human-AI interaction, where the system's reflective capabilities are refined and guided by human expertise. By combining a structured four-state lifecycle with proactive drift detection and a formalized channel for human-in-the-loop review, the framework can ensure that its memory is not only scalable and efficient but also coherent, reliable, and trustworthy over extended periods of operation.

Lifecycle Stage / Concept	Description	Primary Function	Associated Technologies/ Mechanisms
Generated	The initial state for all newly created memory entries.	Marks the beginning of a memory's existence within the system.	Not specified in detail, but involves initial parsing and tagging.
Activated	A state for frequently accessed memories, optimized for fast retrieval.	Improves performance by prioritizing high-utility information.	Loading into KV-cache or fast-access cache ⁶ .
Merged	A state where stable, high-value knowledge is integrated into the parametric base model.	Provides permanent, low-cost access to core, enduring knowledge ⁶ .	Parameter updates, distillation techniques ⁷ .
Archived	A state for low-value, outdated, or redundant information.	Frees up resources and declutters the active memory space ⁶ .	Migration to cheaper storage or deletion.
Insight Drift	Measures the semantic misalignment between an original rationale and its later interpretation ³ .	Quantifies the degradation of memory coherence over time.	Vector similarity analysis, temporal thresholds ³ .
Contextual Entropy	Quantifies the degradation of memory coherence in distributed systems ³ .	Measures the loss of logical consistency across a collaborative network.	Information theory-based metrics on distributed data ³ .
Reflection Interface	A mechanism for human feedback, coherence recovery, and memory updates ³ .	Bridges the gap between automated reasoning and human oversight ³ .	Intelligent queuing, notification systems, human-in-the-loop (HITL) workflows ² .

Efficient Retrieval and Inter-Agent Consistency Protocols

The ultimate value of a memory system is realized through its retrieval processes, which must be both highly efficient to support real-time decision-making and contextually aware to provide relevant information. In a multi-agent collaborative setting, this challenge is compounded by the need for consistent access to a shared memory state. An

effective retrieval system cannot afford to be a brute-force search across all memory types; it must employ intelligent strategies to optimize performance and accuracy. The literature strongly indicates that a hybrid retrieval approach, combining symbolic filtering with semantic search, is the standard for modern systems 6 . Purely semantic search, while powerful for finding similar concepts, lacks the precision to filter by specific attributes like time, source, or access control lists (ACLs). Conversely, purely symbolic filtering is too rigid to handle nuanced queries. The MemOS framework exemplifies a mature solution by having its MemOperator module support hybrid retrieval that leverages both techniques simultaneously 6 . This allows a query to be broken down into structured components (e.g., by tags, time ranges, ACLs) and combined with semantic similarity searches (e.g., vector cosine similarity) to produce a highly targeted and relevant set of results 6 .

Further refinement of retrieval quality comes from adopting a fine-grained, state-aware approach. Traditional experiential memory systems often retrieve experiences based on a coarse, global task or scene context, which can lead to inaccurate decisions because the retrieved experience may not align with the agent's specific current state 4 . The State-Aware Memory (SAMem) paradigm directly addresses this limitation by organizing memory at the level of 'state-specific reasoning thoughts' 4 . When an agent needs to make a decision, it retrieves only the most relevant experience for its precise decision context, focusing on informative cues and avoiding distractions from state-misaligned guidance 4 . Experiments have shown that this approach leads to superior performance and substantially improved task-solving efficiency compared to coarse-grained methods 4 . This principle should be a cornerstone of the proposed framework, ensuring that the information retrieved is not just relevant, but perfectly attuned to the agent's immediate situation. This is complemented by a task-aligned routing mechanism, as proposed by MemOS, which first parses user input into a topic-concept-fact structure to formulate a retrieval strategy before executing the search 6 . This prevents the system from expending resources on irrelevant memory types and is essential for meeting strict latency targets, such as the 250ms insight recall time cited as a target for real-time operation 3 .

While efficient retrieval is crucial for an individual agent, the most significant challenge in a collaborative environment is ensuring multi-agent memory consistency 1 . This refers to the fundamental requirement that all agents share a coherent and up-to-date view of the collective memory state. Without strong consistency guarantees, agents can operate on stale or conflicting information, leading to coordination failures and emergent dysfunctional behaviors. The literature identifies this as the most pressing open challenge, highlighting a critical gap in current research 1 . The MAICC framework,

designed for cooperative MARL, demonstrates one way to manage this by retrieving relevant team-level trajectories combined with individual sub-trajectories, implicitly acknowledging the need to balance shared and private knowledge ⁵. However, a more formal solution is required for a general-purpose AGI framework. This solution must go beyond simple shared-state databases, which are susceptible to race conditions and require complex locking or concurrency control mechanisms ².

The proposed framework must therefore establish standardized protocols for inter-agent consistency. One approach could involve implementing distributed consensus algorithms, akin to those used in distributed databases, to manage updates to shared memory blocks. Alternatively, the framework could adopt a model similar to MemOS's MemStore, which provides a governed interface for memory exchange ⁶. In this model, when an agent wishes to update a shared memory unit, it broadcasts the update through the MemStore. Other agents can then subscribe to changes for specific memory units or categories of interest, pulling the latest version when needed. This pull-based model can reduce contention compared to a push-based model where every update must be immediately synchronized. Crucially, the MemStore interface must be accompanied by robust memory access control mechanisms to define who can read, write, or subscribe to specific memory units, managing permissions and interactions with shared memory spaces ¹. By combining a sophisticated, hybrid retrieval engine with a formally specified protocol for inter-agent consistency, the framework can provide the foundation for reliable, scalable, and intelligent collaboration among AGI agents.

Synthesis and Systemic Integration of the Framework

The culmination of this research is a novel, scalable theoretical framework for artificially generated intelligent memory systems, specifically engineered for multi-agent collaborative decision-making. This framework is not a singular invention but a synthesis of proven concepts from the provided literature, reassembled into a cohesive and operationalizable whole. Its design prioritizes engineering efficiency and scalability over biological plausibility, grounding its principles in computer architecture, distributed systems theory, and advanced machine learning paradigms ¹ ³ ⁶. The framework rests on three foundational pillars: a modular and hierarchical architecture, a dynamic hybrid memory model, and a comprehensive lifecycle management system supported by robust retrieval and consistency protocols. Together, these components create a memory system that is not merely a passive archive but an active, adaptive infrastructure for collective intelligence.

The architectural foundation is a three-layer hierarchical structure (I/O, Cache, Main Memory) designed to manage complexity and performance bottlenecks, drawing from computer architecture principles ¹. This is complemented by a modular internal design, inspired by the CMI Insight Layer, which decomposes the memory system into specialized, interoperable modules for context extraction, indexing, drift monitoring, and regeneration ³. This modularity enhances scalability and maintainability. The core of the framework is its hybrid memory model, which integrates textual, activation-based, and parametric forms of memory ⁶. This integration is managed by a Policy-Aware Scheduler that dynamically promotes or demotes memory units between these forms based on usage patterns, ensuring that knowledge is always in its most appropriate and efficient state ⁶. This addresses the fundamental trade-off between the ease of writing to textual memory and the efficiency of reading from parametric memory ⁷.

The framework's intelligence is manifested in its lifecycle management and evolutionary mechanisms. Every memory unit follows a defined path through a four-state finite state machine: Generated, Activated, Merged, and Archived ⁶. This policy-driven process ensures resource efficiency and long-term coherence. Critically, the framework incorporates a "Drift Monitor" to proactively detect semantic decay in stored knowledge, a concept central to the CMI paradigm ³. This allows the system to flag outdated or contradictory information for human review or automatic regeneration, preventing the propagation of flawed logic—a key requirement for reliable long-term collaboration. For retrieval, the framework employs a hybrid engine combining symbolic and semantic search, enhanced by a task-aligned routing mechanism to optimize query strategies and a state-aware retrieval process to ensure maximum relevance ⁴ ⁶.

Finally, the framework is designed for collaboration through a standardized protocol for multi-agent memory consistency, addressing the most significant open challenge identified in the literature ¹. This is facilitated by a MemStore-like interface that allows agents to publish, subscribe, and distribute memory units in a controlled manner, governed by access control rules ⁶. This entire system is designed to operate within strict performance targets, such as sub-250ms retrieval latency, making it suitable for real-time applications ³. By systematically integrating these components—the hierarchical architecture, the hybrid memory model, the dynamic lifecycle management, the intelligent retrieval processes, and the consistency protocols—the proposed framework provides a comprehensive and theoretically sound foundation for building the next generation of collaborative AGI systems. It moves beyond isolated, agent-centric models to create a shared, evolving, and coherent memory infrastructure that empowers collective intelligence.

Reference

1. [2603.10062] Multi-Agent Memory from a Computer ... <https://arxiv.org/abs/2603.10062>
2. Architecting Multi-Agent AI Systems <https://aishwaryasrinivasan.substack.com/p/architecting-multi-agent-ai-systems>
3. Contextual Memory Intelligence: A Foundational Paradigm ... <https://arxiv.org/html/2506.05370v1>
4. SAMem: State-Aware Memory as a Fine-Grained ... <https://aclanthology.org/2026.findings-acl.722/>
5. Multi-agent In-context Coordination via Decentralized ... <https://ojs.aaai.org/index.php/AAAI/article/view/39394>
6. \titlefontMemOS: A Memory OS for AI System <https://arxiv.org/html/2507.03724v1>
7. A Survey on the Memory Mechanism of Large Language ... <https://dl.acm.org/doi/10.1145/3748302>